Niveau(x) d'enseignement: 7 Numéro EDUQ : 8470896
Centre d'intérêt: 841

Bourjolly, Jean-Marie.

Une approche algorithmique des mathématiques enseignées au niveau collégial / Jean-Marie Bourjolly, collab., Michel Saint-Onge, Daniel Lachance. -- Tracy : Collège de Sorel-Tracy, Direction des services pédagogiques, 1984.
95 p. : graph. ; 25 cm.

Code: 99-8062. Notes bibliogr.: p. 93-95.

ISBN 2-550-07591-9

Descripteur(s): *Mathématiques *Algorithme *Informatique *Développement intellectuel *Méthode d'enseignement *Collégial (Niveau)

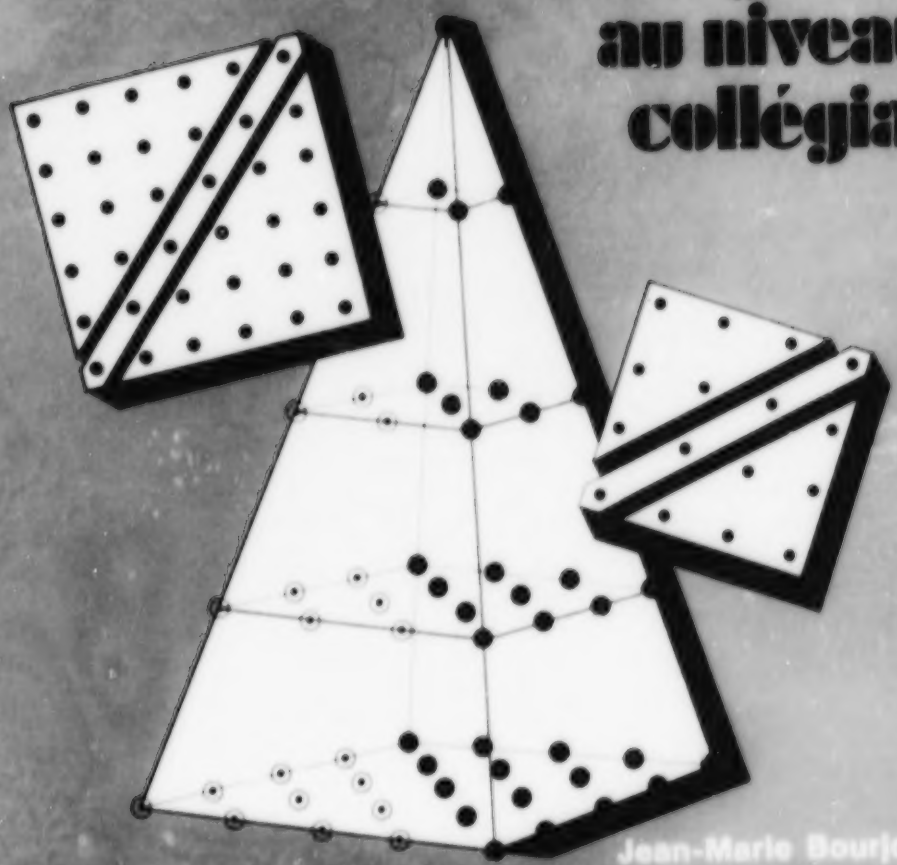
L'arrivée de l'ordinateur dans les écoles secondaires et les collèges remet en question la façon dont les algorithmes sont actuellement enseignés en mathématiques. Le présent ouvrage contient: 1. une réflexion sur le développement cognitif qui conclut qu'il y a un profit à tirer de l'exercice de construction d'algorithmes; 2. une introduction à l'étude des algorithmes; 3. une banque d'algorithmes présents dans l'enseignement des mathématiques; 4. le bref compte rendu d'une expérience-pilote informelle sur le développement de la pensée algorithmique; 5. quelques réflexions quant à une éventuelle implantation de l'approche algorithmique.

Dans la perspective où l'enseignement doit transmettre non seulement un ensemble organisé d'informations mais aussi des habiletés de traitement de ces informations, il apparaît logique de faire une plus large part à la construction d'algorithmes dans l'enseignement des mathématiques; on favorisera ainsi une mutuelle fécondation des mathématiques et de l'informatique. Il faut cependant maintenir l'équilibre entre les différents pôles du développement cognitif et de l'activité mathématique. Dans un plan d'implantation de l'enseignement des mathématiques du point de vue algorithmique, il faudra prendre en considération le contenu des programmes et le temps consacré aux activités de création d'algorithmes.

Type de document: r
Financement: Bénéficiaire - Collège de Sorel-Tracy
Organisme(s) source(s) - MEQ-DGEC
Programme(s) de subvention - PROSIP

EDSQ 8470896

Une approche algorithmique des mathématiques enseignées au niveau collégial



Jean-Marie Bourjolly

Collaborateurs:
Michel Saint-Onge,
Daniel Lachance

EDUQ 8470896

Une approche algorithmique des mathématiques enseignées au niveau collégial

Jean-Marie Bourjolly

Collaborateurs:
Michel Saint-Onge,
Daniel Lachance

Recherche subventionnée par la Direction générale de l'enseignement collégial du Ministère de l'Éducation du Québec dans le cadre du programme de subventions à l'innovation pédagogique (PROSIP).

On peut se procurer des copies de cette publication en en faisant la demande à la
Direction des services pédagogiques
Collège de Sorel-Tracy
3 000, Boul. de la Mairie
Tracy (Québec)
J3R 5V9
(s.v.p. inclure un chèque de 5.00$ par copie commandée)

Achevé d'imprimer à Montréal, octobre 1984
Composition, mise en page et impression: Les Presses Solidaires Inc.

99-8062
I.S.B.N. 2-550-07591-9

Remerciements

Mes remerciements vont tout d'abord à Monsieur Roland Gaudreau, directeur général du C.E.G.E.P. de Sorel-Tracy, qui m'a encouragé à entreprendre ce travail de recherche, suite à un échange de vues sur l'informatique et l'enseignement des mathématiques. Je voudrais remercier également Monsieur Gilles St-Pierre, du Service de la recherche et du développement, au Ministère de l'Éducation du Québec. L'intervention de Monsieur St-Pierre dans le dossier a rendu possible la mise en oeuvre du projet en aplanissant les dernières difficultés d'ordre administratif. Au C.E.G.E.P. de Sorel-Tracy, j'ai bénéficié du soutien de mes collègues Daniel Lachance et Serge Essiambre qui se sont intéressés de près à cetttte recherche. L'aide des Services pédagogiques m'a été très précieuse. Monsieur Jacques Hamel, directeur des services pédagogiques, a manifesté un grand intérêt au travail qui s'est fait. Je désire remercier d'une façon toute spéciale Michel Saint-Onge, le conseiller pédagogique. En plus de contribuer à l'élaboration de la réflexion qui sous-tend ce rapport, il a collaboré étroitement à la rédaction de l'introduction et du chapitre 1. Mes collègues Pierre Désautels (C.E.G.E.P. de Rosemont), Richard Pallascio (C.E.G.E.P. Edouard Montpetit) et Mirette Torkia-Lagacé (C.E.G.E.P. de Limoilou) ont aimablement accepté de me rencontrer et m'ont communiqué des informations utiles à mon travail. Merci à mes amis Janin Jadotte et Roger Lespinasse qui ont relu le manuscrit et m'ont adressé commentaires, critiques et suggestions. Finalement, la réalisation de ce projet n'a été possible que grâce à une importante subvention du Ministère de l'Éducation du Québec (programme P.P.O.S.I.P. d'aide à l'innovation pédagogique).

Table des Matières

Page

Introduction 11

Chapitre Premier
La Problématique

1.1 Quelques observations 15
1.2 Quelques questions 16
1.3 Quelques éléments de réponse 17

Chapitre 2
Algorithmes

2.1 Définition 19
2.2 Algorithmes et informatique 19
2.3 Description d'un algorithme 20
2.4 Efficacité d'un algorithme 20
2.5 Algorithmes et mathématiques 21
2.5.1 Les deux faces de la médaille 21
2.5.2 Influence de l'informatique sur l'enseignement des mathématiques 22
2.5.3 Algorithmes tout faits vs construction d'algorithmes 23
2.5.4 Présentation d'un problème et d'un algorithme de résolution 24

Chapitre 3
Les algorithmes dans l'enseignement collégial

3.1 Résolution d'un système d'équations linéaires 25
3.1.1 Utilisation 25

3.1.2 Mise en situation 26
3.1.3 Connaissances requises 26
3.1.4 Résolution de quelques systèmes simples 26
3.1.5 Redécouverte de la méthode d'élimination de Gauss 27
3.1.6 Autre forme de la méthode de Gauss 31
3.1.7 Méthode d'élimination de Gauss-Jordan 33
3.1.8 Comparaison des algorithmes 3, 3' et 3'' 34
3.1.9 Des confusions à éviter 38
3.2 Calcul de la matrice inverse d'une matrice carrée
3.2.1 Utilisation 38
3.2.2 Connaissances requises 38
3.2.3 Mise en situation 38
3.2.4 Méthode de calcul de A^{-1} 39
3.3 Trouver une famille de vecteurs linéairement indépendants de taille maximale
3.3.1 Énoncé du problème 44
3.3.2 Utilisation 44
3.3.3 Connaissances requises 44
3.3.4 Application 45
3.3.5 Ébauche d'une stratégie 45
3.3.6 Analyse 45
3.3.7 Comment rendre cet algorithme opérationnel? 46
3.4 Programmation linéaire; algorithme du simplexe
3.4.1 Utilisation 46
3.4.2 Connaissances requises 47
3.4.3 Mise en situation 47
3.4.4 Définitions et exemples 47
3.4.5 Histoire de la programmation linéaire 48
3.4.6 La méthode du simplexe: comment cela marche 48
3.4.7 Résumé 59
3.4.8 Retour à la modélisation de problèmes 61
3.4.9 Complexité de la méthode du simplexe 62
3.5 Calculer le déterminant d'une matrice carrée d'ordre n
3.5.1 Utilisation 62
3.5.2 1ère méthode: appliquer la définition 62
3.5.3 2ème méthode: mettre la matrice sous forme échelonnée 62
3.5.4 Méthode ad hoc 63
3.6 Développement du déterminant d'une matrice carrée d'ordre 4
3.6.1 Utilisation 63
3.6.2 Le problème 63
3.6.3 Une méthode 63
3.7 Détermination d'un ensemble de vecteurs linéairement indépendants, de poids maximum
3.7.1 Utilisation 64
3.7.2 Définition du problème 64
3.7.3 Un algorithme de résolution 64
3.8 Arithmétique élémentaire et informatique
3.8.1 Passage de la numération décimale à la numération en base b ($b \neq 10$) 65
3.8.2 Passage de la numération en base b ($b \neq 10$) à la numération décimale 66
3.8.3 La règle de Horner 66
3.8.4 Soustraction par la méthode du complément 66
3.8.5 Algorithme d'Euclide 68
3.8.6 Multiplication égyptienne; algorithme du paysan russe 69
3.8.7 Calcul de X^Y pour deux entiers, X et Y 70

3.9 Calcul approché d'aires; intégration numérique
3.9.1 Utilisation 70
3.9.2 Références 70
3.9.3 Définition du problème 71
3.9.4 Méthode des trapèzes 71
3.9.5 Méthode de la tangente ou du point-milieu 72
3.9.6 Méthode de Simpson 72
3.9.7 Analyse 72
3.9.8 Application: dresser la table de la loi normale 74
3.9.9 Comparaison des trois méthodes 74
3.10 Méthode de recherche dichotomique
3.10.1 Introduction 75
3.10.2 Calcul du logarithme décimal de x 76
3.10.3 Résolution de l'équation f(x) =0 77
3.10.4 Calculer le maximum local d'une fonction 78
3.11 Recherche d'un arbre maximal dans un graphe 79
3.12 Recherche d'un arbre de poids maximum; algorithme de Kruskal 81
3.13 Structure de matroïde; algorithme "vorace" ou "glouton" ("greedy algorithm", en anglais) 82

Chapitre 4
Quelques essais d'implantation de l'approche algorithmique 87

Conclusion 91

Références 93

Introduction

Depuis une trentaine d'années environ, l'enseignement des mathématiques a connu de nombreuses transformations et même, des bouleversements importants. Cela a commencé avec la réforme de la fin des années cinquante qui vit le remplacement — dans la plupart des pays "développés" — des mathématiques "traditionnelles" par les mathématiques dites "modernes". Cette réforme visait à aligner l'enseignement secondaire sur la remise en ordre de l'édifice mathématique qui avait progressivement pris place depuis les travaux de Cantor au dix-neuvième siècle. L'euphorie des premiers temps a fait place à de violentes critiques qui portaient, entre autres choses, sur le fétichisme de la pensée ensembliste et de la méthode axiomatique, le fétichisme de la rigueur et le divorce d'avec la réalité physique, source des idées mathématiques. Aujourd'hui, comme par un retour du balancier, le raisonnement déductif et la rigueur semblent avoir été sacrifiés au profit d'une approche plus intuitive, qui fait une large place à l'induction. Cela se manifeste entre autres par une profusion de problèmes pseudo-réels qui encombrent bon nombre de manuels.

Parallèlement à la réforme des mathématiques "modernes" et de façon plus générale, on a commencé, sous l'influence des travaux de Piaget, à abandonner l'idée que l'habileté à suivre une procédure pré-établie ou à utiliser un algorithme mathématique équivalait à la compréhension d'un problème. Les algorithmes étaient (et sont encore) principalement enseignés comme des procédures à connaître en vue d'une utilisation presque machinale. Mettant en doute le fait que l'habileté à manipuler un algorithme mathématique indique la compréhension ou la maîtrise des concepts mathématiques sous-jacents, on a abandonné la si large part consacrée à l'enseignement du calcul pour chercher plutôt à développer les concepts mathématiques. L'option prise alors fut de privilégier les concepts au détriment des algorithmes, de favoriser la compréhension plutôt que la mémorisation. Cela s'est traduit par l'utilisation d'objectifs d'enseignement vagues tels que "initier...", "familiariser..." et "développer...". Ces objectifs d'enseignement sont difficilement traduisibles en objectifs d'apprentissage et,

lorsqu'ils le sont, cela donne lieu à de très grandes variations d'un enseignant à un autre, ce qui fait qu'on ne peut plus être certain que les étudiants ont acquis les préalables nécessaires pour aborder certains sujets nouveaux. Alors qu'il est facile de mesurer la connaissance d'un théorème ou la compréhension de l'algorithme de résolution d'un certain type d'équations, comment savoir si on est parvenu à "initier les élèves au processus de mathématisation" ou à "développer leur aptitude à penser de façon logique"?

Cette question est directement liée à celle du développement des activités cognitives supérieures. On sait que certaines habiletés intellectuelles ne sauraient être acquises à l'intérieur d'une session de quinze semaines et qu'au contraire, elles ne peuvent être acquises qu'à long terme. Tout enseignement poursuit à la fois deux types d'objectifs. Les premiers sont de type phénotypique, c'est-à-dire qu'ils sont observables dès que le comportement se modifie. Les autres objectifs sont d'ordre génotypique, c'est-à-dire qu'ils concernent des changements profonds de la structure cognitive. Ces changements sont longs, se font sur une longue période de temps et conséquemment, sont difficilement observables par un enseignant au cours d'une brève période. L'évaluation de l'atteinte de tels objectifs échappe

> "...aux moyens primitifs d'évaluation objective dont on dispose actuellement. La tentative extrême d'exprimer toutes les acquisitions par des mesures objectives renverse souvent l'ordre dans le processus didactique consistant à se poser les objectifs et chercher les moyens du contrôle et de l'évaluation de leur réalisation. Ce renversement s'exprime par la tendance à poser les objectifs de l'éducation mathématique en fonction des outils de contrôle et d'évaluation dont on dispose. De tels procédés ont été développés avant tout sur la base du behaviorisme. De ce point de vue, on définit les objectifs spécifiques par des listes détaillées des performances les plus simples, qui définissent de même le programme minimum de l'apprentissage et qui peuvent être évaluées localement dans des situations simples, par des tests standardisés adéquats. Le rapport du NACOME (National Advisory Committee on Mathematical Education, "Overview and analysis of school mathematics grades K-12", Washington, D.C., Conference Board of the Mathematical Sciences, 1975) analyse les conséquences aux États-Unis de la mise en oeuvre dans la réalité scolaire d'une telle conception des objectifs. On observe que, très souvent, l'effort des étudiants et des élèves se concentre sur la maîtrise de performances étroites, isolées, définies par les listes mentionnées. Le programme minimal réduit à ces performances entraîne aussi le minimalisme concernant l'organisation des activités mathématiques variées; il ne suggère pas de situations ouvertes, de recherche libre faite par les élèves, etc... Les auteurs du rapport soulignent que l'apprentissage correct de "mathématiques véritables" exige l'initiation des élèves aux procédés heuristiques, qui ne peuvent pas être décomposés a priori en une suite de petits exercices" [31].

L'enseignement des mathématiques doit-il se plier aux exigences de l'évaluation objective, ou doit-il se construire à partir des exigences du développement intellectuel?

Les moyens raffinés d'évaluation des habiletés et des attitudes mentales, "de toutes ces acquisitions de l'intelligence qui devraient être considérées par la pédagogie des mathématiques comme les finalités de l'apprentissage", font encore défaut.

C'est dans ce contexte que les ordinateurs se préparent à faire une entrée massive et remarquée dans l'enseignement secondaire et collégial. Cela soulève des problèmes inédits. Va-t-on traiter les mathématiques et l'informatique comme deux disciplines complètement cloisonnées? Va-t-on, au contraire, les ouvrir l'une à l'autre, de façon féconde? En quoi la présence des ordinateurs peut-elle influer sur l'enseignement des mathématiques? Quelle est l'importance des mathématiques pour un étudiant en informatique? En quoi les ordinateurs sont-ils utiles à l'acquisition d'habiletés cognitives supérieures? Une première réponse à ces questions nous est suggérée par les considérations suivantes:

1) Les algorithmes se situent au coeur même de l'informatique et à la charnière de cette disci-

pline et des mathématiques (voir les sections 2.2 et 2.5).

2) Un enseignement qui ignore la construction d'algorithmes est un enseignement incomplet. Au contraire, la construction d'algorithmes "constitue une excellente base pour acquérir un raisonnement rigoureux" (voir le chapitre 1 et la section 2.5).
3) dans l'enseignement des mathématiques, les algorithmes sont généralement présentés comme du tout-cuit plutôt que comme des objets à construire. Cela correspond à un objectif, souvent inconscient ou implicite, qui consiste à favoriser l'habileté à résoudre un problème particulier plutôt que le développement de l'habileté intellectuelle de résolution de problèmes.

L'arrivée de l'ordinateur dans les écoles secondaires et les collèges remet en question la façon dont les algorithmes sont enseignés en mathématiques; elle rend possible le renversement de la tendance plus haut signalée (voir section 2.5) pour développer les habiletés cognitives des étudiants. En faisant une plus large part à la construction d'algorithmes dans l'enseignement des mathématiques, non seulement on s'inscrit dans la perspective d'une meilleure formation des étudiants, mais aussi on favorise une mutuelle fécondation des mathématiques et de l'informatique.

Cette recherche avait d'abord pour but d'explorer la possibilité d'enseigner les mathématiques au niveau collégial d'un point de vue algorithmique et de dégager les implications pédagogiques d'un tel changement de perspective. Il s'agit d'une réflexion amorcée il y a quatre ans à l'Université de Waterloo et qui s'est développée tout au long des douze mois que cette étude a duré.

Cet ouvrage contient
— une réflexion sur le développement cognitif, qui conclut à la nécessité pour tout enseignement de transmettre non seulement un ensemble organisé d'informations, mais aussi des habiletés de traitement de ces informations. Cela suggère pour le moins que l'on a avantage à ne pas s'en tenir à l'enseignement d'algorithmes tout faits et qu'il y a un profit à tirer de l'exercice de construction d'algorithmes.
— une introduction à l'étude des algorithmes et un examen de leur rôle en mathématiques et en informatique.
— une banque d'algorithmes présents dans l'enseignement des mathématiques au niveau collégial. La présentation suggérée vise à favoriser l'analyse et la compréhension des concepts sous-jacents et à donner un sens à la démarche de construction d'un algorithme.
— le compte-rendu d'une expérience-pilote informelle axée sur le développement de la "pensée algorithmique".
— quelques réflexions quant à une éventuelle implantation de l'approche algorithmique.

Chapitre 1

La Problématique

L'enseignement étant du domaine de la pratique, c'est l'observation de son effet sur l'apprentissage des élèves qui est la principale source du questionnement pédagogique. Aussi c'est sur la base de difficultés d'étudiants de niveau collégial que nous allons entreprendre notre réflexion.

1.1 Quelques observations

L'enseignement n'a de sens que par les apprentissages qu'il suscite. Par ailleurs, ces apprentissages prennent tout leur sens s'ils se situent dans la ligne d'un développement. C'est pourquoi tout enseignant s'attend à ce que les élèves qu'il reçoit à un niveau donné aient certaines caractéristiques, maîtrisent certaines habiletés et soient munis des connaissances jugées préalables à l'enseignement qu'il doit leur dispenser.

Il est assez courant de constater que les élèves qui arrivent au C.E.G.E.P. ont de plus en plus de difficulté à répondre à ces attentes des enseignants. L'intérêt suscité par la recherche de Mirette Torkia-Lagacé [50] sur l'accession des élèves arrivant dans les collèges au stade de pensée formelle, témoigne de cette préoccupation.

Plutôt que de montrer comment les élèves s'éloignent des caractéristiques d'un niveau de développement donné, nous allons tout simplement relever certaines observations qui nous semblent décrire un certain niveau de développement conceptuel en mathématiques. Ces observations ne sont pas rigoureusement contrôlées. Cependant, si elles rejoignent celles que d'autres professeurs ont pu faire en enseignant, nous aurons avec eux un terrain commun pour chercher à comprendre comment nous pourrions modifier la situation présente en considérant les mathématiques d'un point de vue algorithmique.

Plusieurs élèves semblent avoir une conception purement fonctionnelle des mathématiques. Pour eux, il s'agit d'appliquer une méthode ou une formule préalablement apprise pour arriver à la "bonne réponse". Ils considèrent la connaissance d'algorithmes et de formules comme le principal objectif de l'apprentissage en mathématiques. De nombreux élèves n'en sont même pas là. Ils appliquent les maigres résultats qu'ils ont pu mémoriser à tort et à travers et se soucient très peu de justifier les opérations qu'ils font, comme si toute cette gymnastique était au fond dépourvue de signification. Faut-il alors s'étonner que les étudiants soient souvent incapables d'entreprendre des démarches longues de résolution de problèmes? Ils n'arrivent pas à donner un sens à ce qu'ils font. Ils n'ont pas de questionnement mathématique. Ils n'ont que des réponses à des tâches brèves.

La difficulté qu'ils éprouvent à construire des stratégies de résolution de problèmes semble étroitement liée à leur conception des mathématiques et de ce qu'est une preuve mathématique. En effet, pour bon nombre d'étudiants, la preuve se réduit à une simple vérification sur un ou deux exemples numériques de l'exactitude d'une formule ou du bon fonctionnement d'un algorithme. Chaque exemple, chaque cas et, à un autre niveau, chaque théorème est traité individuellement, sans référence à des modèles abstraits plus généraux ou à des théories générales. Comme les connaissances sont atomisées au lieu d'être structurées, on observe un phénomène courant de rétention à court terme et une difficulté de transfert très grande.

Il n'est pas étonnant dans ce contexte de constater un manque d'intérêt pour les problèmes mathématiques et pour le raisonnement. Ceci se traduit par une difficulté grave à utiliser un langage précis et des définitions rigoureuses, et à imaginer des démarches diverses pour la résolution de problèmes.

Évidemment, ces difficultés conduisant au manque d'intérêt, cela ne peut que se traduire en termes d'abandons ou d'échecs fréquents.

1.2 Quelques questions

Alors que l'enseignement des mathématiques — suite aux bouleversements des années cinquante et soixante — se veut orienté vers l'acquisition de concepts plutôt que vers la mémorisation à outrance de méthodes et de formules, comment se fait-il que les élèves arrivant dans les collèges soient tellement axés sur l'application de recettes à la résolution de problèmes courts?

Il est en effet étonnant de constater un niveau d'abstraction aussi faible quand on s'attendrait au contraire à une conceptualisation supérieure à celle obtenue antérieurement.

N'y a-t-il pas un lien entre ce faible niveau d'abstraction et l'intérêt pour les démarches courtes de résolution de problèmes?

L'élève incapable de comprendre des processus de raisonnement sur des réalités abstraites peut-il réussir son cours, s'en sortir autrement qu'en mémorisant des séquences d'opérations?

Lorsque l'enseignement vise d'autres buts que la connaissance de formules et l'application d'algorithmes et que l'élève n'a ni les connaissances préalables, ni les habiletés intellectuelles nécessaires, peut-il prendre intérêt aux démarches qui lui sont proposées?

Comment en est-on venu à développer si peu l'esprit mathématique, la capacité d'abstraction, l'intérêt pour les stratégies de résolution de problèmes? Comment se fait-il que les mathématiques perdent leur sens pour les élèves? Pourquoi ne s'intéressent-ils pas aux constructions de l'esprit?

1.3 Quelques éléments de réponse

Dans[25], les auteurs insistent sur le fait que l'enseignement des sciences doit être différent de l'enseignement des mathématiques. Or, plusieurs professeurs cherchent à enseigner leurs disciplines comme les mathématiques. Ils cherchent à "prouver" quelque chose.

> "Quand les sciences sont enseignées de cette façon, ce qui arrive très souvent dans les écoles où l'on considère les scientifiques et les mathématiciens comme interchangeables, on aboutit à la négation même de la science." ([25], p. 24)

On peut se demander si le mouvement inverse n'est pas également vrai. Lorsqu'on cherche à enseigner les mathématiques de façon empirique, en mettant l'accent sur la démarche inductive au détriment de la démarche déductive, n'aboutit-on pas également à la négation des mathématiques?

Les mathématiques sont un langage que les scientifiques emploient largement à cause de son haut degré d'abstraction.

> "La mathématique joue (là) un rôle privilégié pour l'intelligence de ce que nous nommons le réel, réel physique comme réel social. Notre mathématique sécrète, par nature, l'économie de pensée et, par là, permet seule de classer, de dominer, de synthétiser parfois en quelques brèves formules un savoir qui, sans elle, finirait par ressembler à quelque fâcheux dictionnaire encyclopédique infiniment lourd". ([12] cité dans [45])

Les mathématiques sont également une construction logique. La démarche déductive permet — en s'appuyant sur un système d'axiomes et sur les règles de la logique — de prouver des résultats de façon rigoureuse. La déduction logique "est le vrai et l'unique moteur de la pensée mathématique" ([17], p. 17)

L'enseignement qui ne respecte pas les caractéristiques de la discipline enseignée ne convient pas à cette discipline! L'approche empirique des mathématiques pourrait expliquer les résultats obtenus car il est probable que les concepts mathématiques ne se développent que par l'observation des opérations logiques des mathématiques elles-mêmes.

L'enseignement des mathématiques devrait donc, comme tout autre enseignement, respecter ses deux pôles: la structure de ses connaissances et les processus par lesquels elles ont été acquises. L'enseignement ne peut se réduire à une simple communication d'un corps de connaissances. Il doit transmettre également les processus par lesquels ces connaissances sont acquises. Tout comme le processus par lequel Einstein est arrivé à la formulation du principe de la relativité est loin d'être apparent dans les équations qui l'expriment, de même, le processus par lequel le mathématicien arrive à la formulation d'un algorithme n'apparaît pas dans l'énoncé de cet algorithme.

De nombreux enseignants ne distinguent pas ces deux aspects de l'apprentissage: l'acquisition d'une structure de connaissances et l'acquisition des processus de traitement cognitif. Aussi, ils croient que les habiletés cognitives sont innées, qu'il suffit de faire apprendre des symboles, des algorithmes et que les élèves "intelligents" sauront bien les traiter, les utiliser pour résoudre des problèmes.

Nous croyons que la pensée est formée non seulement d'un ensemble organisé d'informations mais également d'habiletés de traitement de ces informations et que ces capacités mentales ne fonctionnent pas indépendamment de l'objet sur lequel elles opèrent. L'enseignement ne peut viser un seul de ces deux pôles. L'acquisition d'un contenu ne peut se faire véritablement si on désire qu'il vienne augmenter la structure de signification qu'un individu utilise, sans l'acquisition d'habiletés de traitement de cette information. Ces deux pôles sont indissociables. L'élève ne peut pas ne pas avoir à mémoriser les éléments d'information, tout comme il ne peut pas ne pas avoir à traiter ces informations en vue de les rendre réutilisables.

L'enseignement d'un algorithme comme information sans qu'il y ait enseignement des processus conduisant à sa conception, tout comme l'enseignement des processus de résolution de problèmes sans en dégager des algorithmes à mémoriser, sont des enseignements incomplets. Ce processus d'alternance entre les deux pôles — structure d'information et procédure de traitement — caractérise l'enseignement. C'est grâce à lui que s'opère le développement cognitif. Les éléments d'information prennent ainsi, avec la diversité de leur usage dans des processus de traitement variés, une signification de plus en plus riche.

On peut représenter le développement cognitif comme une spirale dont le mouvement est suscité par les deux pôles: structure d'information et processus de traitement.

Structure d'information

Processus de traitement

Dans le contexte spécifique des mathématiques, ce développement est aussi le fruit d'une recherche continuelle d'équilibre entre

l'automatisme et la compréhension;
le concret et l'abstrait;
l'induction et la déduction;
l'intuition, d'une part, le formalisme et la rigueur, d'autre part;
la pratique des applications et la maîtrise des concepts et des structures;
l'utilisation et la construction d'algorithmes...

L'enseignement des mathématiques considère-t-il les habiletés intellectuelles comme un objet d'apprentissage, ou les considère-t-il uniquement comme un préalable? Est-ce qu'il respecte l'équilibre entre les différents pôles de l'activité mathématique? L'algorithme en mathématiques est-il utilisé à la fois comme élément de la structure d'information et comme processus de traitement ou, au contraire, d'une façon qui privilégie l'un de ces aspects au détriment de l'autre?

Ces questions renvoient à l'orientation actuelle de l'enseignement des mathématiques et la réponse à ces questions nous semble expliquer les résultats obtenus jusqu'à présent.

Chapitre 2

Algorithmes

2.1 Définition

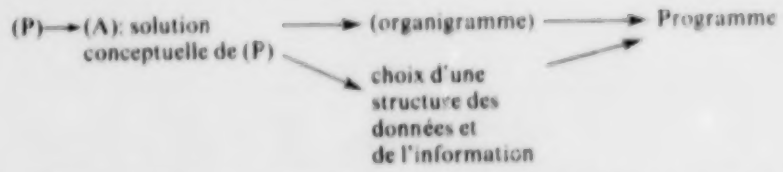
Le terme "algorithme" donne lieu à des définitions diverses et il n'y a même pas de consensus quant à son étymologie. Le concept d'algorithme est à rapprocher d'autres concepts tels que méthode, recette, procédure... En ce sens, un algorithme est une méthode de résolution d'un problème qui ramène l'accomplissement d'une tâche donnée à un ensemble de "sous-tâches" spécifiques qui doivent être exécutées séquentiellement, selon un ordre déterminé.

Il ressort de cette définition que:
a) "algorithme" se trouve associé à "problème" et à "solution".
b) La nature du problème en question n'est pas spécifiée.
c) un algorithme présente un caractère opérationnel et constructif.

2.2 Algorithmes et Informatique

"L'étude des algorithmes se situe au coeur même de la science informatique" affirment d'entrée de jeu Aho, Hopcroft et Ullman [1]; "Algorithms + Data Structures = Programs", tel est le titre d'un classique de la littérature informatique [57]. Knuth précise: "Un programme consiste à formuler un algorithme dans un langage bien défini (Fortran, Cobol, Pascal, ...). Ainsi un programme d'ordinateur représente un algorithme, mais l'algorithme lui-même est un concept mental qui existe indépendamment de toute représentation. D'une manière similaire le concept du nombre 2 existe dans notre esprit, même quand il n'est pas écrit" [29]. Ajoutons qu'un organigramme est une représentation (graphique) d'un algorithme et qu'il se situe entre ce dernier et le programme qui lui correspond. Tout programme d'ordinateur est

l'aboutissement d'un processus séquentiel qui commence avec un problème (P) donné et qui passe par un algorithme de résolution (A).



Une fois l'algorithme décrit de façon précise dans les moindres détails et avec une structure de données appropriée, l'écriture du programme se trouve réduite à un simple exercice technique. On peut donc dire qu'un programme vaut ce que valent d'algorithme qu'il représente et la structure de données choisie.

2.3 Description d'un algorithme

Le choix d'une structure de données correspond à une façon particulière de décrire un algorithme. "Souvent il existe plus d'une façon de décrire un algorithme donné; les descriptions qui visent à clarifier les concepts sous-jacents sont souvent très différentes de celles qui donnent lieu à des implantations efficaces sur ordinateurs"[10]. Ainsi, à la section 3.4, on utilise ce que l'on appelle des dictionnaires pour représenter les programmes linéaires et c'est dans ce contexte que la méthode du simplexe est exposée. "Les dictionnaires fournissent un outil commode pour expliquer les principes de base de l'algorithme du simplexe. Cependant, quand il s'agit d'implanter cette méthode sur ordinateur pour résoudre des problèmes de grande taille, les considérations d'efficacité des calculs et de précision numérique l'emportent sur les finesses d'ordre didactique"[10].

Il s'agit donc d'adopter la représentation qui convient le mieux à chaque situation. Ainsi les algorithmes étudiés en classe doivent être — autant que possible — présentés d'une manière qui en favorise la compréhension.

2.4 Efficacité d'un algorithme

Généralement, un problème peut être résolu par plusieurs algorithmes. Il se pose alors la question de la comparaison et du choix. Soit, par exemple, à calculer le déterminant d'une matrice carrée d'ordre n (Section 3.5). Si on effectue ce calcul par la méthode de réduction de Gauss, il requiert au plus $t_1(n) = \frac{n\,(n+1)\,(2n+1)}{6}$ additions et multiplications, plus $\frac{n\,(n+1)}{2}$ divisions. Si au contraire on applique brutalement la définition du déterminant, on peut avoir à effectuer $t_2(n) = n\,!\,(n-1)$ multiplications, plus $(n\,!-1)$ additions.

Les nombres t_1 et t_2 représentent des **bornes supérieures** du **nombre maximum** d'opérations requises par chacun des deux algorithmes pour calculer le déterminant d'une matrice quelconque. Ils sont obtenus de façon théorique (voir section 3.5) et ils mesurent l'efficacité (on dit aussi la complexité) des deux algorithmes en question par rapport à deux critères, celui de la vitesse d'exécution et celui du "pire-des-cas". Si on se base sur t_1 et t_2, c'est sans conteste le 1er algorithme qui sera choisi. En effet, excepté pour les valeurs $n = 2$ et $n = 3$, la fonction t_1 est systématiquement dominée par t_2, la distance entre ces deux fonctions devenant rapidement sidérale. Si on avait eu $t_1(n) < t_2(n)$ pour $n \leqslant 100$ et $t_1(n) > t_2(n)$ pour $n > 100$ par

exemple, le choix de l'algorithme n'aurait pas été le même pour un problème de petite taille que pour un problème de grande taille.

À côté de la mesure de la vitesse d'exécution d'un algorithme suivant le critère du pire-des-cas, il existe aussi des mesures d'efficacité moyenne. Ces mesures sont généralement empiriques, basées sur le comportement observé de l'algorithme. Il arrive qu'un mauvais algorithme selon le critère du pire-des-cas se révèle excellent en pratique. C'est le cas de la méthode du simplexe (section 3.4). La question de l'efficacité d'un algorithme est de première importance pour les informaticiens. Il ne sert pas à grand chose — sur le plan pratique — de concevoir un programme (donc un algorithme) dont le temps d'exécution est démesurément grand. Cela dit, un tel algorithme peut malgré tout présenter un grand intérêt théorique ou didactique. C'est le cas, par exemple, de la méthode de Cramer pour résoudre un système de n équations linéaires à n variables.

2.5 Algorithmes et Mathématiques

Nous avons tous appris à l'école des algorithmes qui nous permettent d'effectuer les opérations d'addition, de soustraction, de multiplication et de division, d'extraire la racine carrée d'un nombre, de calculer le plus grand commun diviseur ou le plus petit commun multiple de deux nombres, ... On peut dire avec Engel [21] que "les algorithmes ont toujours joué un grand rôle dans les mathématiques scolaires". On peut même affirmer sans crainte de se tromper qu'ils jouent un grand rôle à tous les niveaux des mathématiques. Pourtant, combien d'étudiants en mathématiques de niveau collégial savent ce que c'est qu'un algorithme? Combien sont capables de justifier la méthode qu'ils utilisent quand ils effectuent la division — voire même l'addition de deux nombres?

La deuxième question renvoie au fait que l'enseignement des algorithmes se réduit généralement à la mémorisation et à l'application de recettes. Étant donné que l'essence des mathématiques réside dans l'analyse et la résolution de problèmes plutôt que dans l'application machinale d'un ensemble de règles, il s'ensuit que les algorithmes — à cause de la façon dont ils sont enseignés — se trouvent passablement discrédités.

Il faut bien les enseigner puisqu'ils sont utiles, mais on n'en fait pas grand cas!

2.5.1 Les deux faces de la médaille

Distinguons
— le processus de création d'un algorithme
— la justification mathématique d'un algorithme
— l'exécution d'un algorithme.

Alors que la création et la justification d'un algorithme constituent une activité enrichissante, tout-à-fait dans l'esprit des mathématiques, l'exécution d'un algorithme présente un caractère machinal, dépourvu d'intérêt, et peut être confiée à une machine.

Idéalement, il faudrait se concentrer sur le processus de création et l'opération de justification mathématique, l'exécution (automatique) des algorithmes servant à ajuster le tir, c'est-à-dire à en vérifier l'exactitude sur des exemples choisis ou à fournir des indications quant à des améliorations possibles. La réalité est tout autre: les algorithmes sont enseignés comme des recettes qu'il s'agit de mémoriser et d'exécuter machinalement. On demande aux étudiants de faire le travail de la machine, activité fastidieuse et peu formatrice; le travail de réflexion et d'analyse est réduit au minimum; l'évaluation de l'apprentissage porte presque exclusivement sur la rétention d'informations et, de façon plutôt exceptionnelle, sur la maîtrise d'habiletés cognitives.

Évidemment, quand on ne dispose pas d'une calculatrice programmable ou d'un ordinateur, on est bien obligé d'accorder plus de place qu'on ne le voudrait à l'exécution d'algorithmes dans l'enseignement. Il est même possible que cet exercice en vienne à prendre toute la place, à cause, entre autres choses, de contraintes de temps. De plus, la question de l'efficacité se pose alors très peu en pratique: quand il s'agit de calculer le déterminant d'une matrice d'ordre 2 ou 3 à la main, on utilise la première méthode qui nous vient à l'esprit; s'il s'agit d'une matrice d'ordre 4 ou 5, l'algorithme le plus efficace paraît encore désespérément lent. Qui s'est jamais risqué à calculer le déterminant d'une matrice d'ordre 6 ou plus ne présentant aucune structure aisément exploitable?

Quand il s'agit d'expliquer comment l'enseignement des algorithmes en est venu à sacrifier l'essentiel, Engel opine: "L'absence d'un outil efficace [ordinateur, calculatrice programmable] pour utiliser [les algorithmes] interdisait une attitude franchement "algorithmique". Jusqu'à l'avènement de l'ordinateur, on n'admettait pas que la construction d'algorithmes puisse présenter de l'intérêt et constituer, comme elle le fait, une excellente base pour acquérir un raisonnement rigoureux, qui est indispensable dans tous les domaines de la culture" [21].

2.5.2 Influence de l'informatique sur l'enseignement des mathématiques

L'apparition des calculatrices de poche a eu une certaine influence sur les mathématiques scolaires, notamment dans le domaine du calcul trigonométrique et celui des fonctions logarithmiques et exponentielles.

L'introduction des micro-ordinateurs dans les écoles offre l'occasion de promouvoir la "pensée algorithmique" dans l'enseignement des mathématiques. Cela consiste à apprendre aux étudiants à analyser et à créer des algorithmes, l'exécution de ceux-ci — ou plutôt des programmes correspondants — étant confiée aux ordinateurs. L'étudiant et l'enseignant se trouvent libérés des calculs fastidieux et peuvent se concentrer sur les concepts et les résultats mathématiques qui sous-tendent l'algorithme.

Dans [21], sous le titre "Les Mathématiques Scolaires dans l'Optique Algorithmique", Engel écrit:

> "Au début du siècle, F. Klein entreprit de réformer l'enseignement des mathématiques. Son mouvement prit pour devise: "la pensée fonctionnelle". Les réformateurs assuraient que "la pensée fonctionnelle" devait être instaurée à tous les niveaux de l'enseignement mathématique. Ce que l'étudiant en mathématiques devait retenir de ses études, c'était de penser en termes de fonctions. Cette réforme modifia profondément les mathématiques scolaires.
> Le temps est venu d'une nouvelle réforme dont la devise serait la "pensée algorithmique". Le raisonnement par algorithmes doit sous-tendre tout l'enseignement mathématique. Il faut que les enseignants se rendent compte que "une des activités principales de la classe doit être de créer et d'analyser des algorithmes. De nombreux concepts peuvent être introduits de façon constructive par des programmes. Chaque programme est l'expression d'un concept, d'une fonction, d'un objet mathématique. Les programmes sont des instruments abstraits qu'il est possible de manipuler et dont on peut étudier les "sorties" pour les utiliser de façon diverse en "entrées". En observant la structure et le comportement d'un programme, l'étudiant apprend de façon active le concept que celui-ci représente. L'interaction entre l'élève et le programme est décisive pour le processus d'acquisition des connaissances. L'étudiant doit apprendre à modifier les programmes et à élaborer des programmes complexes à partir de programmes simples. Clarté et simplicité sont les grandes règles du bon style et des bonnes mathématiques. Les algorithmes doivent être clairs. Ceci vaut autant pour les organigrammes que pour les instructions. Au début, les deux types de représentation s'équilibrent. Il faut progressivement réduire la part des organi-

grammes, outils primitifs et rudimentaires dont on ne peut plus se passer si l'on s'en sert trop.

Au départ, l'élève écrira ses algorithmes dans son langage propre. Puis, étape par étape, on lui demandera de le normaliser, pour aboutir en fin de compte à un vocabulaire restreint *(if-then-else, while-do, do-until, etc.)* facilement traduisible en un langage informatique quelconque. Ce vocabulaire ressemble aux pièces d'un jeu de meccano avec lesquelles on peut construire de nombreux dispositifs différents (instruments abstraits ou algorithmes). Les ordinateurs ont pour but de faire fonctionner ces instruments comme c'est celui du courant électrique de faire marcher un aspirateur...

Pour toute fonction étudiée en classe, il est nécessaire de donner au moins un algorithme permettant de la calculer. Pour associer un sens concret à f, l'élève doit connaître un algorithme qui *sans difficulté* donne f (x) pour une entrée x. On disposera, de préférence, de plusieurs algorithmes pour la même fonction, de manière à pouvoir les comparer et identifier: le plus efficace, le plus clair, le plus élémentaire, le plus universel".

2.5.3 Algorithmes tout faits vs. construction d'algorithmes

Dans ce qui précède, on a opposé les activités qui visent à développer les habiletés d'analyse et de création d'algorithmes à la méthode traditionnelle qui consiste à mémoriser et à exécuter des algorithmes. On peut, plus généralement, opposer le processus de construction d'une solution à un problème au fait de prendre connaissance d'une solution toute faite. Dans un cas, le mécanisme de la recherche et de la découverte se trouve démonté; on apprend à être autonome; on apprend, face à un problème, à énoncer des hypothèses, à ébaucher une stratégie, à revoir ses hypothèses, à raffiner sa stratégie au contact d'exemples concrets... Dans l'autre cas, on reçoit une information. Celle-ci peut s'avérer utile dans un contexte semblable mais on ignore comment on y est parvenu.

Il va de soi que la première approche est autrement plus formatrice que la seconde. Toutefois, cette distinction est artificielle pour une bonne part, la réalité étant bien plus complexe que cela. Pour comprendre, analyser, construire, il faut des connaissances préalables. Parfois l'on se trouve dans l'obligation de transmettre des informations aux étudiants alors que ceux-ci sont dépourvus des connaissances nécessaires pour les analyser ou en comprendre la justification. C'est ce qui arrive quand on enseigne les algorithmes d'addition et de multiplication à de jeunes enfants. On leur transmet une information; on s'évertue à développer chez eux des automatismes. Cette information et ces premières habiletés constituent les fondations de l'édifice. Ce sont des connaissances qui servent de base à l'acquisition d'autres connaissances, d'autres habiletés, et il n'est pas question — à ce stade-là — de justifier ces algorithmes. De même il existe des algorithmes qu'il est difficile d'enseigner à un collégien moyen autrement que sous la forme d'une transmission d'information. Ce n'est donc pas une affaire de tout ou rien. Il faut recourir à la justification mathématique, à l'analyse, aux activités de création chaque fois que c'est possible, mais la maîtrise des habiletés reliées à la construction d'algorithmes passe aussi par la lecture et la mémorisation d'algorithmes tout faits. D'autre part, une fois qu'un algorithme est construit, il devient une information qui doit être mémorisée de façon organisée en vue d'une utilisation ultérieure.

> "La lecture, au niveau de la langue naturelle a autant d'importance que l'écriture (la rédaction). L'art d'écrire s'acquiert par la lecture de textes en prose bien écrits (étude de la littérature). De même, il faut accorder autant d'importance à la lecture qu'à l'écriture des programmes. L'art de rédiger des programmes s'acquiert par la lecture de nombreux programmes bien rédigés (étude de la littérature algorithmique). Dans les petites classes, on fera surtout lire et exécuter des algorithmes en demandant aux élèves de trouver ce à quoi ils servent. Pour rédiger un programme élaboré, il faut, comme pour faire une bonne dissertation, une certaine maturité et un certain sens de l'organisation." [21]

2.5.4 Présentation d'un problème et d'un algorithme de résolution.

Compte tenu de l'analyse qui précède, il y aurait avantage à présenter une solution algorithmique comme suit:

1. Connaissances requises

Faire l'inventaire des concepts, définitions et résultats mathématiques dont la connaissance est nécessaire à l'analyse et à la résolution du problème. Rappeler ces notions au besoin.

2. Position du problème

3. Analyse

Analyser le problème; établir des liens avec d'autres problèmes, avec les objets mathématiques définis en 1.

4. Ébauche de solution

Élaborer une ébauche de stratégie à l'aide de quelques idées simples. Préciser un certain nombre de questions qu'on se propose de clarifier en cours de route.

Partir au besoin d'un ou deux exemples numériques. Leur appliquer la stratégie minimale et noter les points de désaccord entre cette stratégie et les problèmes numériques en question.

5. Raffiner la stratégie

Analyser les résultats obtenus en 4.

Réappliquer 4 au besoin, avec de nouvelles hypothèses, une stratégie plus fine, de nouveaux exemples.

Analyser la complexité de la méthode mise au point; la simplifier, si possible; la comparer avec d'autres méthodes.

6. Algorithme final

L'énoncer.

L'exécuter.

En étudier la complexité.

Le comparer avec d'autres algorithmes.

Chapitre 3

Les algorithmes dans l'enseignement collégial

Dans ce chapitre se trouvent rassemblés un certain nombre d'algorithmes de résolution de problèmes de niveau collégial. Les quatre premiers problèmes relèvent de l'algèbre linéaire et sont traités en détail, selon le modèle présenté à la section 2.5.4. Les sections 3.1-3.7 pourraient servir d'ossature à un cours d'algèbre linéaire (Math 105 et Math 205) enseigné selon l'approche algorithmique. Les sept problèmes de la section 3.8 appartiennent à l'arithmétique élémentaire. Les quatre premiers d'entre eux sont étudiés dans tous les cours d'introduction à l'informatique. Les trois autres constituent des exercices à la fois élégants et formateurs.

La section 3.9 offre une comparaison de trois algorithmes classiques d'intégration numérique. La section 3.10 regroupe trois problèmes qui peuvent se résoudre par une méthode de recherche dichotomique. Le problème de la section 3.11 représente un cas particulier de celui de la section 3.12; de la même façon, le problème de la section 3.3 est un cas particulier de celui de la section 3.7. Ce dernier, ainsi que le problème de la section 3.12 sont eux-mêmes des cas particuliers d'un problème plus général, traité à la section 3.13.
Les sections 3.10 et 3.13 visent à illustrer le fait qu'il y a des classes d'algorithmes et des problèmes de même type, appelant des solutions semblables.

Note
Les numéros de cours **soulignés** sont ceux au programme desquels le problème étudié est spécifiquement mentionné (cf. **Les Cahiers de l'Enseignement Collégial, 1983-1986**; Math 105 désigne le cours 201-105-77; Info 101, le cours 420-101-82, etc...).

3.1 Résolution d'un système d'équations linéaires

3.1.1 Utilisation

Math 105, Math 122, Math 171, Math 205

3.1.2 Mise en Situation

Proposer trois ou quatre problèmes dont la solution passe par la définition, puis par la résolution d'un système d'équations linéaires.

Dans chaque cas, le problème initial se trouve ramené à celui-ci:
Comment résoudre le système d'équations qui vient d'être défini?

3.1.3 Connaissances requises

Les règles élémentaires de l'algèbre.
Savoir résoudre l'équation $ax = b$.

Il est utile de rappeler aux étudiants comment résoudre l'équation $ax = b$.

3.1.4 Résolution de quelques systèmes simples

Proposer aux étudiants trois ou quatre systèmes faciles à résoudre (systèmes écrits sous forme échelonnée), de façon à leur mettre l'eau à la bouche et à mettre en branle le processus de réflexion.

Quelques exemples:

(a)
$$\begin{aligned} x_1 - x_2 + 2x_3 &= 5 \\ x_2 - x_3 - 3x_4 &= 2 \\ x_3 + x_4 &= 1 \\ x_4 &= 2 \end{aligned}$$

(b)
$$\begin{aligned} x_2 + 3x_4 + x_5 &= 3 \\ x_3 + 2x_4 + x_5 &= 1 \\ x_1 - x_5 &= -4 \end{aligned}$$

(c)
$$\begin{aligned} x_1 - x_2 + 2x_3 - x_4 - x_5 + x_6 &= 5 \\ x_3 - x_5 &= 1 \\ x_5 - x_6 &= 0 \\ x_6 &= 4 \end{aligned}$$

(d)
$$\begin{aligned} x_1 - x_2 - x_4 &= -1 \\ x_2 - 2x_3 + 2x_6 &= 0 \\ x_4 + 3x_5 - x_6 &= 2 \\ x_7 &= 3 \end{aligned}$$

Les buts poursuivis sont les suivants:

— ***convaincre les étudiants de ce que ces exemples ne présentent aucune difficulté conceptuelle.***

— ***les inviter à rechercher ce qu'ils ont en commun (la structure...) et qui rend simple leur résolution. Pour les aider à se faire une opinion, on peut leur demander d'essayer de résoudre un système dense, comme celui-ci:***

(e)
$$\begin{aligned} x_1 - x_2 + 2x_3 - 3x_4 &= -8 \\ -x_1 + x_2 - 3x_3 + 4x_4 &= 10 \\ -3x_1 - x_2 + x_3 - x_4 &= 0 \\ 2x_1 + x_2 + 2x_3 - 5x_4 &= -11 \end{aligned}$$

— *leur faire toucher du doigt que le problème de la résolution d'un système d'équations linéaires est pratiquement réglé quand le système est écrit sous forme échelonnée (exemples (a) – (d)).*

On est donc ramené au problème suivant:

Étant donné un système (S) d'équations linéaires, est-il possible de définir un système (S') écrit sous forme échelonnée et ayant le même ensemble-solution que (S)? Dans l'affirmative, comment s'y prendre?

3.1.5 Redécouverte de la méthode d'élimination de Gauss

Il est préférable de procéder du particulier au général, de partir d'un exemple numérique choisi de façon que les règles d'élimination s'imposent "naturellement".

Considérons le système (S) défini dans l'exemple (e) ci-dessus. On va essayer de trouver un système (S') qui soit à la fois
(i) équivalent à (S), c'est-à-dire ayant le même ensemble-solution, et
(ii) écrit sous forme échelonnée.

Stratégie envisagée: Procéder par étapes. Définir successivement des systèmes (S_1), (S_2), ..., $(S_r) = (S')$ qui soient tous équivalents à (S) et écrits sous une forme de plus en plus proche de la forme échelonnée.

— Partant de (S), on peut définir un système (S_1) dont la 2e équation ne contient pas de terme en x_1. Il suffit pour cela de la définir comme la somme, membre à membre, des deux premières équations de (S).

$$(S_1)\qquad \begin{aligned} x_1 - x_2 + 2x_3 - 3x_4 &= -8 \\ -x_3 + x_4 &= 2 \\ -3x_1 - x_2 + x_3 - x_4 &= 0 \\ 2x_1 + x_2 + 2x_3 - 5x_4 &= -11 \end{aligned}$$

(S_1) semble représenter un pas dans la bonne direction, mais est-il équivalent à (S)? Il l'est! En effet, si les égalités $y = a$ et $z = b$ sont simultanément satisfaites, il en est de même des égalités $y = a$ et $y + z = a + b$ et vice-versa.

— La création d'un système (S_2) dont les équations 3 et 4 sont dépourvues de terme en x_1 peut s'effectuer en deux étapes:
(i) créer deux équations auxiliaires en multipliant chaque terme de l'équation 1 de (S_1) par 3 [resp. par -2]
(ii) ajouter membre à membre la 1ère équation auxiliaire [resp. la 2ème équation auxiliaire] obtenue en (i) à l'équation 3 [resp. à l'équation 4] de (S_1).

$$(S_2)\qquad \begin{aligned} x_1 - x_2 + 2x_3 - 3x_4 &= -8 \\ -x_3 + x_4 &= 2 \\ -4x_2 + 7x_3 - 10x_4 &= -24 \\ 3x_2 - 2x_3 + x_4 &= 5 \end{aligned}$$

(S_2) est équivalent à (S_1). En effet,
(i) on ne change pas l'ensemble-solution d'une équation donnée quand on multiplie chaque terme (y compris le terme constant) par un même nombre, **non nul**. Ainsi, les égalités $y = a$ et $k \cdot y = k \cdot a$ $(k \neq 0)$ sont simultanément satisfaites, ou aucune ne l'est.

(ii) Les systèmes d'égalités suivants sont simultanément satisfaits, ou aucun ne l'est.

$$\begin{cases} y = a \\ z = b \end{cases} \quad ; \quad \begin{cases} ky = ka \ (k \neq 0) \\ z = b \end{cases} \quad ; \quad \begin{cases} ky = ka \\ ky + z = ka + b \end{cases} \quad (k \neq 0)$$

$$\begin{cases} y = a \\ ky + z = ka + b \ (k \neq 0) \end{cases}$$

— Le système (S_3) présente une difficulté nouvelle. Si l'on utilise l'équation 1 pour éliminer le terme en x_2 de l'équation 3 ou de l'équation 4 selon la procédure utilisée précédemment, on y réintroduit du même coup un terme en x_1. Cette difficulté peut être contournée en permutant les équations 2 et 3 ou les équations 2 et 4. En permutant les équations 2 et 3 de (S_2) on obtient un système (S_3) équivalent à (S_2) parce que l'ensemble-solution d'un système d'équations linéaires est invariant par permutation de ces dernières.

$$(S_3) \qquad \begin{aligned} x_1 - x_2 + 2x_3 - 3x_4 &= -8 \\ -4x_2 + 7x_3 - 10x_4 &= -24 \\ -x_3 + x_4 &= 2 \\ 3x_2 - 2x_3 + x_4 &= 5 \end{aligned}$$

— En divisant chaque terme de l'équation 2 par -4 on obtient une équation équivalente dont le 1er terme non nul est égal à 1. Cela nous amène au système équivalent (S_4)

$$(S_4) \qquad \begin{aligned} x_1 - x_2 + 2x_3 - 3x_4 &= -8 \\ x_2 - \tfrac{7}{4}x_3 + \tfrac{5}{2}x_4 &= 6 \\ -x_3 + x_4 &= 2 \\ 3x_2 - 2x_3 + x_4 &= 5 \end{aligned}$$

— En poursuivant dans la même veine, nous obtenons (S_5), (S_6), (S_7) et finalement, (S')

$$(S_5) \qquad \begin{aligned} x_1 - x_2 + 2x_3 - 3x_4 &= -8 \\ x_2 - \tfrac{7}{4}x_3 + \tfrac{5}{2}x_4 &= 6 \\ -x_3 + x_4 &= 2 \\ \tfrac{13}{4}x_3 - \tfrac{13}{2}x_4 &= -13 \end{aligned}$$

$$(S_6) \qquad \begin{aligned} x_1 - x_2 + 2x_3 - 3x_4 &= -8 \\ x_2 - \tfrac{7}{4}x_3 + \tfrac{5}{2}x_4 &= 6 \\ x_3 - x_4 &= -2 \\ \tfrac{13}{4}x_3 - \tfrac{13}{2}x_4 &= -13 \end{aligned}$$

$$(S_7) \qquad \begin{aligned} x_1 - x_2 + 2x_3 - 3x_4 &= -8 \\ x_2 - \tfrac{7}{4}x_3 + \tfrac{5}{2}x_4 &= 6 \\ x_3 - x_4 &= -2 \\ -\tfrac{13}{4}x_4 &= -\tfrac{13}{2} \end{aligned}$$

$$(S') \qquad \begin{aligned} x_1 - x_2 + 2x_3 - 3x_4 &= -8 \\ x_2 - \tfrac{7}{4}x_3 + \tfrac{5}{2}x_4 &= 6 \\ x_3 - x_4 &= -2 \\ x_4 &= 2 \end{aligned}$$

Au cours de la résolution de cet exercice, trois règles (appelées opérations élémentaires) sont apparues. Il est nécessaire de les verbaliser et de les mémoriser avec précision. On sera constamment appelé à s'en servir par la suite.

On demande aux étudiants d'appliquer ces opérations à trois ou quatre exemples numériques et d'indiquer à chaque étape quelle opération est utilisée, et comment. Cela fait, ils sont mûrs pour passer à la phase de généralisation. On peut les convaincre de ce que ces trois opérations suffisent en général à mettre un système quelconque sous forme échelonnée une fois qu'ils ont compris à quoi sert chacune de ces opérations et dans quelles circonstances elle est utilisable.

Cela nous amène à la formulation d'un algorithme, lequel se présente comme l'aboutissement d'un processus d'apprentissage.

Algorithme 1 (Mettre un système d'équations linéaires sous forme échelonnée)

Données du problème
Un système (S) de m équations linéaires à n variables, $x_1, x_2, \ldots, x_n$.
Chaque équation est "ordonnée": le terme en x_i est suivi du terme en x_{i+1}, etc...

Résultats
Un système (S') de m équations à n variables, équivalent à (S) et écrit sous forme échelonnée.

Appliquer la procédure qui suit à l'équation 1, puis à l'équation 2, ..., et finalement, à l'équation m.

Supposons que l'équation i soit en cours d'examen ($1 \leqslant i \leqslant m$). Soit (T), le système formé des équations i, i + 1, i + 2, ... , m.

$$(T)\qquad \begin{aligned} a_{ii}x_i &+ a_{i2}x_2 + \ldots + a_{in}x_n = b_i \\ a_{i+1,i}x_i &+ a_{i+1,2}x_2 + \ldots + a_{i+1,n}x_n = b_{i+1} \\ &\cdots\cdots\cdots\cdots\cdots\cdots \\ a_{mi}x_i &+ a_{m2}x_2 + \ldots + a_{mn}x_n = b_m \end{aligned}$$

Étape 1 (Localiser le pivot)
Soit $N = \{ t /$ il existe au moins un indice $s \geqslant i$ pour lequel a_{ti} n'est pas nul $\}$. Soit j, le plus petit élément de N.
Si $a_{ij} = 0$, on définit $M = \{ s \mid s \geqslant i$ et $a_{si} \neq 0 \}$.
Soit k un élément quelconque de M. On permute les indices des équations i et k, i.e. la i^{Θ} équation devient la k^{Θ}, et vice-versa.
On dit que x_i est la **variable de base** correspondant à l'équation i. On dit aussi que a_{ii} est le **pivot** de l'équation i; c'est le 1er coefficient non nul de cette équation.

Étape 2 (La valeur du pivot est mise à 1)
Diviser chaque terme de l'équation i par a_{ii}.

Étape 3 (Éliminer x_i des équations i + 1, i + 2, ... , m)
Pour chaque indice $s > i$ tel que $a_{si} \neq 0$, transformer l'équation s comme suit:
(i) Définir une équation auxiliaire en multipliant chaque terme de l'équation i par $-a_{si}$
(ii) ajouter membre à membre l'équation auxiliaire obtenue en (i) à l'équation s. Cela définit la nouvelle équation s.

Fin de la procédure.

À ce stade, les étudiants ont déjà expérimenté la "substitution à rebours" sur les exemples (a)

et (c), c'est-à-dire qu'ils ont porté la valeur de x_4 [resp. x_6] *dans la 3ème équation de façon à pouvoir calculer* x_3 [resp. x_5]; *puis ils ont porté la valeur de* x_3 et x_4 [resp. x_5 et x_6] *dans la 2ème équation, etc... Cette procédure est formalisée comme suit:*

Algorithme 2 (Substitution à rebours)

Données du problème
Un système d'équations linéaires (S') sous forme échelonnée. (S') possède r équations à n variables, avec $r \leqslant n$. Dans chaque équation il existe au moins une variable dont le coefficient n'est pas nul.

Résultats
L'ensemble-solution de (S').

Soit $x_{p(1)}$ la variable de base correspondant à l'équation 1, ..., $x_{p(r)}$, la variable de base correspondant à l'équation r. Les autres variables sont dites **hors-base.**

Étape 1 (Expression des variables de base en fonction des variables hors-base)
Dans chaque équation, faire passer les termes relatifs aux variables hors-base dans le membre de droite. On obtient un système équivalent (S'') où les variables de base s'expriment en fonction des variables hors-base.

Étape 2 (Détermination de l'ensemble-solution)
Les variables hors-base peuvent prendre des valeurs arbitraires.
La valeur de $x_{p(r)}$ se lit dans l'équation r de (S''). Pour calculer $x_{p(r-1)}$, on remplace $x_{p(r)}$ par sa valeur dans l'équation $r - 1$ de (S'').
Pour calculer $x_{p(r-2)}$, on remplace $x_{p(r-1)}$ et $x_{p(r)}$ par leur valeur dans l'équation $r - 2$ de (S'').
Et ainsi de suite...

Fin

On peut résoudre un système d'équations linéaires en combinant les algorithmes 1 et 2.

Algorithme 3 (Résolution d'un système d'équations linéaires par la méthode d'élimination de Gauss)

Données du problème
Un système (S) de m équations linéaires à n variables

Résultats
L'ensemble-solution de (S)

Étape 1 (Mettre le système sous forme échelonnée)
Appliquer Algorithme 1 à (S). On obtient un système équivalent (T).

Étape 2
Si (T) possède une équation de la forme $0 \cdot x_1 + \ldots + 0 \cdot x_n = b$ avec $b \neq 0$, on conclut que (S) ne possède pas de solution (Son ensemble-solution est vide). Autrement, aller à l'Étape 3.

Étape 3 (Substitution à rebours)
Éliminer toutes les équations de la forme $0 \cdot x_1 + \ldots + 0 \cdot x_n = 0$. Soit (S'), le système ainsi obtenu. Appliquer Algorithme 2 à (S').

Fin.

Remarque
Dans la plupart des manuels d'Algèbre Linéaire, on s'empresse dès le début de définir la matrice augmentée d'un système d'équations linéaires. Les opérations élémentaires effectuées sur des équations sont alors présentées comme des opérations sur les lignes d'une

matrice. L'allégement de l'écriture qui en résulte est contrebalancé par le fait que la justification des opérations élémentaires sur les lignes d'une matrice oblige à un incessant va-et-vient mental entre un système donné d'équations et sa matrice augmentée. Le passage d'un mode de représentation à un autre est une excellente chose en soi mais le danger est grand qu'il ne se fasse pas et que l'on se trouve réduit, de ce fait, à appliquer aveuglément des recettes. C'est pourquoi il est sans doute préférable d'attendre que les étudiants aient bien compris à la fois la méthode et sa justification avant d'introduire la notion de matrice augmentée et d'adapter les algorithmes 1, 2 et 3 à ce nouveau contexte.

3.1.6 Autre forme de la Méthode de Gauss

Les algorithmes 1, 2 et 3 existent aussi sous d'autres formes.
Considérons le système (T) défini dans l'exemple (e) du paragraphe précédent. La 1ère équation de (T) s'écrit

(1) $$x_1 = -8 + x_2 - 2x_3 + 3x_4$$

En remplaçant x_1 par sa valeur dans les équations 2, 3 et 4 de (T), on obtient un système (T_1) de 3 équations à 3 variables d'où x_1 est absent.

$$(T_1) \qquad \begin{aligned} -x_2 + x_4 &= 2 \\ -4x_2 + 7x_3 - 10x_4 &= -24 \\ 3x_2 - 2x_3 + x_4 &= 5 \end{aligned}$$

La 2ème équation de (T_1) s'écrit

(2) $$x_2 = 6 + \tfrac{7}{4}x_3 - \tfrac{5}{2}x_4$$

En remplaçant x_2 par sa valeur dans l'équation 3 de (T_1) on obtient un système (T_2) de deux équations à 2 variables, x_3 et x_4

$$(T_2) \qquad \begin{aligned} -x_3 + x_4 &= 2 \\ \tfrac{13}{4}x_3 - \tfrac{13}{2}x_4 &= -13 \end{aligned}$$

La 1ère équation de (T_2) s'écrit

(3) $$x_3 = -2 + x_4$$

En remplaçant x_3 par sa valeur dans l'équation 2 de (T_2) on obtient un système (T_3)

$$(T_3) \qquad -\tfrac{13}{4}x_4 = -\tfrac{13}{2}$$

(T_3) contient une équation linéaire à une variable que nous sommes en mesure de résoudre (voir section 3.1.3). On a donc ramené la résolution d'un système de 4 équations linéaires à 4 variables à celle d'une équation linéaire à 1 variable. Voyons cela de plus près. Les systèmes suivants sont équivalents:

$$(T) \quad \begin{cases} (1) \\ (T_1) \end{cases} \quad \begin{cases} (1) \\ (2) \\ (T_2) \end{cases} \quad \begin{cases} (1) \\ (2) \\ (3) \\ (T_3) \end{cases} \quad \begin{cases} (1) \\ (2) \\ (3) \\ (4): x_4 = 2 \end{cases}$$

L'équation (T_3) a pour solution

(4) $$x_4 = 2$$

En remplaçant x_4 par sa valeur dans (3) (substitution à rebours) on obtient à nouveau une

équation linéaire à une variable, x_1. On continue ainsi jusqu'à la résolution complète du système.

Il est facile de voir que $\begin{cases}(1)\\(T_1)\end{cases}$ correspond à (S_2),

$\begin{cases}(1)\\(2)\\(T_2)\end{cases}$ correspond à (S_3), $\begin{cases}(1)\\(2)\\(3)\\(T_3)\end{cases}$ correspond à (S_4) et finalement,

$\begin{cases}(1)\\(2)\\(3)\\(4)\end{cases}$ correspond à (S').

Exercice
(i) En suivant les calculs effectués dans ce paragraphe, montrer que (T) et $\begin{cases}(1)\\(2)\\(3)\\(4)\end{cases}$ sont équivalents.

(ii) Écrire la méthode utilisée dans ce paragraphe sous la forme d'un algorithme qu'on appellera Algorithme 1'.

(iii) Comparer étape par étape les algorithmes 1 et 1' en les appliquant à un ou deux exemples numériques.

En examinant les exemples (a) – (d), on se rend compte qu'il est plus immédiat de résoudre (b) et (d) que (a) et (c). (b) et (d) sont sous **forme échelonnée réduite**, c'est-à-dire que si x_i est la variable de base correspondant à l'équation i, le coefficient de x_i dans les équations 1,2, ..., $i-1$ est nul. Il n'y a pas de substitution à rebours à effectuer dans ce cas: il suffit de faire passer les variables hors-base dans le membre de droite de chaque équation. La forme échelonnée réduite présente un certain intérêt d'ordre théorique et elle s'obtient aisément à partir de la forme échelonnée. La description que l'on vient de donner des systèmes (b) et (d) suggère en effet ce qui suit:

Algorithme 2' (Mettre un système échelonné sous forme échelonnée réduite)

Données du problème
système de r équations linéaires à n variables, écrit sous forme échelonnée.

Résultats
un système équivalent de r équations à n variables écrit sous forme échelonnée-réduite.

Appliquer la procédure qui suit aux équations r, puis $r-1, \ldots, 1$

Supposons que l'équation i ($1 \leqslant i \leqslant r$) soit en cours d'examen et que x_i soit la variable de base correspondante.
On élimine x_i des équations 1, 2, ..., $i-1$:
Pour chaque indice $s < i$ tel que $a_{si} \neq 0$, transformer l'équation s comme suit:
(i) Définir une équation auxiliaire en multipliant chaque terme de l'équation i par $-a_{si}$
(ii) Ajouter membre à membre l'équation auxiliaire obtenue en (i) à l'équation s. Cela définit la nouvelle équation s.

Fin de la procédure

Cela nous amène à modifier l'algorithme 3.

Algorithme 3'

Données
Résultats } Les mêmes que pour l'algorithme 3

Étape 1
Étape 2 } cf. Algorithme 3

Étape 3

Éliminer les équations de la forme $0x_1 + \ldots + 0x_n = 0$
Soit (S'), le système ainsi obtenu. Appliquer Algorithme 2' à (S')
Dans chaque équation, faire passer les termes relatifs aux variables hors-base dans le membre de droite.
Soit (S'') le système équivalent ainsi obtenu. Les variables hors-base prennent des valeurs arbitraires.
La valeur des variables de base se lit directement dans (S'').

Fin

Exercice
Comparer étape par étape les algorithmes 2 et 2' en les appliquant à un ou deux exemples numériques.

3.1.7 Méthode d'élimination de Gauss-Jordan

Cette méthode consiste à effectuer **dans le même temps** les opérations de l'algorithme 2' et celles de l'étape 3 de l'algorithme 1.

Algorithme 3'' (Résolution d'un système d'équations linéaires par la méthode d'élimination de Gauss-Jordan)

Données
Résultats } Les mêmes que pour l'algorithme 3

Appliquer la procédure qui suit à l'équation 1, puis à l'équation 2, ..., et finalement, à l'équation m.

Supposons que l'équation i soit en cours d'examen ($1 \leqslant i \leqslant m$).

Étape 1 (Localiser le pivot)
Étape 2 (Mettre la valeur du pivot à 1) } (cf. Algorithme 1)

Étape 3 (Éliminer x_i des équations 1, 2, ... , i − 1, i + 1, ... , m)
Pour chaque indice $s \neq i$ tel que $a_{si} \neq 0$, transformer l'équation s conformément à l'étape 3 de l'algorithme 1.

Fin de la procédure

3.1.8 Comparaison des algorithmes 3, 3' et 3"

Exercice
Appliquez les algorithmes 3, 3' et 3" à l'exemple (e) de la section 3.1.4. Comptez dans chaque cas le nombre d'additions, de multiplications et de divisions effectuées.
Essayez de déterminer pourquoi Algorithme 3" requiert un plus grand nombre d'additions et de multiplications.

Cet exercice a pour but d'attirer l'attention des étudiants sur l'efficacité relative des algorithmes, 3, 3' et 3" et donc, de les préparer à ce qui suit.

Soit (S), un système de n équations à n variables. (Pour simplifier les calculs, on suppose que le nombre d'équations est égal au nombre de variables et que (S) ne possède qu'une seule solution). On peut analyser la vitesse d'exécution des algorithmes 3, 3' et 3" en calculant le nombre maximum d'opérations nécessaires à la résolution de (S) quand on utilise l'un ou l'autre de ces algorithmes.

Tableau 1. Complexité de Algorithme 1

Pour éliminer la variable	Additions	Multiplications	Divisions
x_1	$(n-1)\,n$	$(n-1)\,n$	n
x_2	$(n-2)\,(n-1)$	$(n-2)\,(n-1)$	$n-1$
x_k	$(n-k)\,(n-k+1)$	$(n-k)\,(n-k+1)$	$n-k+1$
x_{n-1}	2	2	2
x_n	0	0	1

Tableau 2. Complexité de Algorithme 2

Calcul de x_t par substitution	Additions	Multiplications
$t = n$	0	0
$t = n-1$	1	1
$t = n-2$	2	2
$t = 2$	$n-2$	$n-2$
$t = 1$	$n-1$	$n-1$

Tableau 3. Complexité de Algorithme 2'

Élimination de x_t des équations 1, 2, ..., $t-1$	Additions	Multiplications
$t = n$	$n-1$	$n-1$
$t = n-1$	$n-2$	$n-2$
$t = 2$	1	1
$t = 1$	0	0

Tableau 4. Complexité de Algorithme 3"

Pour éliminer la variable	Additions	Multiplications	Divisions
x_1	$(n-1)\,n$	$(n-1)\,n$	n
x_2	$(n-1)(n-1)$	$(n-1)(n-1)$	$n-1$
x_k	$(n-1)(n-k+1)$	$(n-1)(n-k+1)$	$n-k+1$
x_{n-1}	$(n-1)\,2$	$(n-1)\,2$	2
x_n	$(n-1)\,1$	$(n-1)\,1$	1

Le nombre maximum d'opérations nécessaires à l'exécution des algorithmes 1, 2, 2', 3, 3' et 3" est donné par:

$$v_1 = \sum_{k=1}^{n} (n-k)(n-k+1) \text{ additions} + \sum_{k=1}^{n} (n-k)(n-k+1) \text{ multiplications}$$

$$+ \sum_{k=1}^{n} k \text{ divisions}$$

$$v_2 = v_2' = \sum_{k=1}^{n} (k-1) \text{ additions} + \sum_{k=1}^{n} (k-1) \text{ multiplications}$$

$$v_3 = v_1 + v_2$$

$$v_3' = v_1 + v_2' = v_1$$

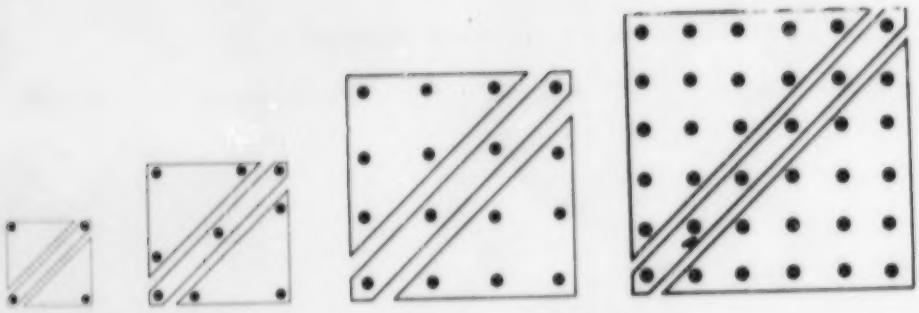
$$v_3'' = (n-1) \sum_{k=1}^{n} k \text{ additions} + (n-1) \sum_{k=1}^{n} k \text{ multiplications} + \sum_{k=1}^{n} k \text{ divisions}$$

Une légère digression

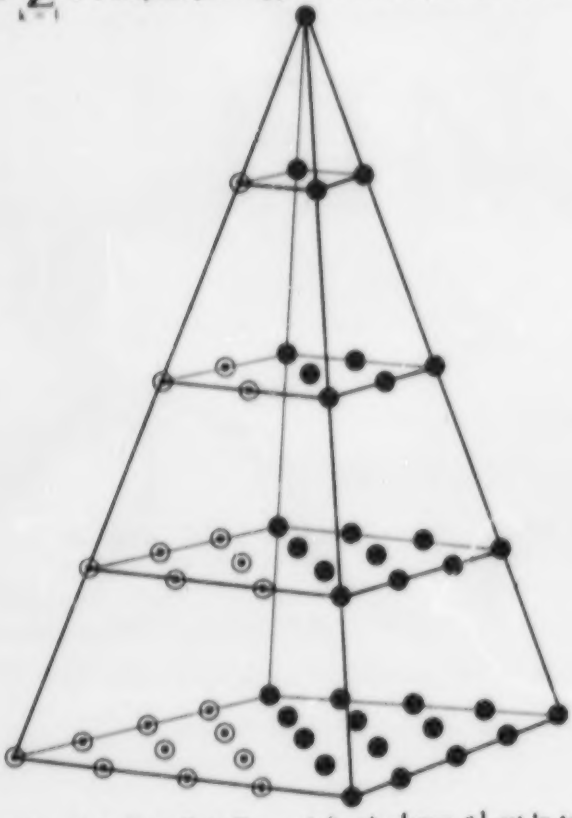
Il est important, chaque fois que cela est possible, de lier des disciplines mathématiques différentes, de façon à décloisonner les connaissances des étudiants et à encourager le transfert d'habiletés d'une discipline à une autre.

L'évaluation de v_1, v_2, ..., v_3'' *offre l'occasion de rappeler aux étudiants comment calculer* $\sum_{k=1}^{n} k$ *et* $\sum_{k=1}^{n} k^2$ *(ou de le leur apprendre, s'ils ne le savaient déjà). Pour cela, on peut utiliser, outre les méthodes algébriques bien connues, des méthodes figuratives qui ont le mérite de capter l'attention des étudiants, de soutenir leur intérêt et d'offrir aux plus audacieux d'entre eux un vaste champ d'investigations.*

Ainsi, $\sum_{k=1}^{n} k$ *se calcule aisément grâce aux constructions suivantes:*



Pour calculer $\sum_{k=1}^{n} k^2$ *, on peut faire appel à une construction pyramidale.*



On voit que $1^2 + 2^2 + 3^2 + 4^2 + 5^2 = \alpha + \beta$, où α [resp. β] est la somme des boules creuses [resp. pleines].

$$\alpha = 1 + 3 + 6 + 10 \qquad \beta = 1 + 3 + 6 + 10 + 15$$

Les valeurs de α et de β se lisent dans le triangle de Pascal.

```
1
1 1
1 2 1
1 3 3 1
1 4 6 4 1
1 5 10 10 5 1
1 6 15 20 15 6 1
1 7 21 35 35 21 7 1
```

$\alpha = 20$ et $\beta = 35$

Plus généralement, $\sum_{k=1}^{n} k^2 = \binom{n+1}{n-2} + \binom{n+2}{n-1}$

(voir [49] et [54]).

Après évaluation de ces expressions, il vient:

$$v_1 = \frac{n(n+1)(2n+1)}{6} \text{ additions} + \frac{n(n+1)(2n+1)}{6} \text{ multiplications} + \frac{n(n+1)}{2} \text{ divisions}$$

$$v_2 = v_2' = \frac{n(n-1)}{2} \text{ additions} + \frac{n(n-1)}{2} \text{ multiplications}$$

$$v_3'' = \frac{(n-1)n(n+1)}{2} \text{ additions} + \frac{(n-1)n(n+1)}{2} \text{ multiplications} + \frac{n(n+1)}{2} \text{ divisions}$$

Quand n prend des valeurs très grandes, le terme de degré le plus élevé de chaque expression l'emporte sur les autres. Ainsi, v_2 est négligeable par rapport à v_1 pour de grandes valeurs de n. *(On a l'occasion, cette fois, de faire le lien avec le cours de MATH 103 en calculant la limite de ces expressions quand n devient infiniment grand.)*

Algorithme 3 et Algorithme 3' requièrent un nombre d'additions et de multiplications de l'ordre de $\frac{n^3}{3}$ tandis que Algorithme 3" en requiert de l'ordre de $\frac{n^3}{2}$.

Le critère du **nombre maximum** d'opérations requises nous permet de classer ces algorithmes comme suit: Algorithme 3 et Algorithme 3' sont équivalents. Ils requièrent le même nombre d'opérations arithmétiques. Le second est toutefois **plus opérationnel** en ce sens qu'il exprime l'élimination des variables sous une forme plus directement programmable. Algorithme 3" est plus lent que les 2 autres et présente de ce fait un intérêt moindre.

Remarques

1) Quand on compare Algorithme 3' et Algorithme 3" on voit que seul l'ordre de certaines opérations change. Dans un cas, toutes les éliminations s'effectuent dans le même temps tandis que dans l'autre, elles s'effectuent en deux étapes distinctes. Cette différence — apparemment mineure — a des conséquences importantes quant à l'efficacité de l'algorithme.

2) Il existe d'autres critères de comparaison tels que le **nombre moyen** d'opérations requises, la précision numérique des résultats, etc...
3) Pour mauvaise que soit une méthode en pratique, **elle peut** présenter des avantages d'ordre théorique qui en justifient l'étude. Ainsi en est-il, par exemple, de la méthode de Cramer, qui est bien plus lente que la méthode de Gauss, en plus d'être de portée moins générale.

Exercice (pour les étudiants qui connaissent la méthode de Cramer)
(i) En quoi la méthode de Cramer est-elle de portée moins générale que la méthode de Gauss?
(ii) Calculer le nombre maximum d'opérations arithmétiques requises par la méthode de Cramer. Comparer la vitesse d'exécution de cette méthode à celle de la méthode de Gauss.

3.1.9 Des confusions à éviter

Dans [10] Vašek Chvátal écrit: "Beaucoup d'étudiants ne sont pas avertis de la distinction entre la méthode d'élimination gaussienne et celle de Gauss-Jordan, sans compter qu'ils ignorent souvent que la première est supérieure à la seconde. (Il n'est pas nécessaire de rechercher bien loin la source de cette confusion: il existe au moins un manuel de programmation linéaire qui décrit la méthode de Gauss-Jordan sous le nom d'élimination gaussienne; au moins trois autres livres présentent la méthode de Gauss-Jordan sans même mentionner celle de Gauss, laquelle est pourtant meilleure)." Cette confusion, on la retrouve dans la plupart des textes d'algèbre linéaire.

3.2 Calcul de la Matrice Inverse d'une Matrice Carrée

Référence: [7]

3.2.1 Utilisation

Math 105, Math 205

3.2.2 Connaissances Requises

a) Matrices: Définition, addition de deux matrices, produit de deux matrices, produit d'une matrice par un nombre réel, propriétés de ces opérations.
b) Résolution d'un système d'équations linéaires par la méthode de Gauss; matrice élémentaire; expression d'une opération élémentaire comme le produit à gauche par une matrice élémentaire; équivalence-ligne de deux matrices; matrice augmentée d'un système d'équations linéaires.
c) Définition de la matrice inverse d'une matrice carrée.
d) **Théorème 1**
Si une matrice carrée A possède une inverse à gauche C et une inverse à droite B (i.e. si $CA = AB = I$), alors elle est inversible et $B = C = A^{-1}$
Corollaire 1 Si A est inversible, son inverse est unique.
e) **Théorème 2**
Si A et B sont deux matrices carrées de même ordre et si elles sont inversibles, alors $A \cdot B$ est inversible et son inverse est égale à $B^{-1} \cdot A^{-1}$
Corollaire 2: Si $A_1, A_2, \ldots, A_k$ sont inversibles, alors il en est de même de leur produit, et $(A_1 \ldots A_k)^{-1} = A_k^{-1} \ldots A_2^{-1} \cdot A_1^{-1}$

3.2.3 Mise en situation

Imaginons une machine à coder/décoder des nombres. Chaque fois qu'on lui donne une suite

de 100 nombres, elle leur fait subir une certaine transformation et elle restitue une autre liste de 100 nombres. Le 1er nombre de la liste sortante c'est la valeur codée (ou décodée) du 1er nombre de la liste entrante, etc... On suppose que le codage ne laisse place à aucune ambiguïté, c'est-à-dire qu'à une liste entrante il correspond une liste sortante et une seule. Deux correspondants peuvent communiquer à l'aide de deux machines, l'une servant de codeur et l'autre de décodeur. Le correspondant 1 introduit une liste X dans son **codeur**. Il en ressort une liste Y qui est communiquée au correspondant 2. Celui-ci introduit la liste Y dans son **décodeur** et il en sort X.

Maintenant, supposons que le codeur [resp. le décodeur] fonctionne comme suit. On y introduit préalablement une matrice carrée d'ordre 100, A [resp. B]. Par la suite, chaque fois qu'on lui fournit un vecteur X [resp. Y] de dimension 100, la machine calcule $A \cdot X$ [resp. $B \cdot Y$] et crache un vecteur X' [resp. Y'] d'ordre 100. Le vecteur Y que l'on fournit au décodeur est égal à $X' = AX$. on veut s'assurer que $Y' = BY = BAX$ est bien égal à X. BAX est toujours égal à X dans le cas où $B \cdot A = I$. Comme les messages circulent dans les deux sens, on voudrait aussi qu'un message codé sur le 2ème appareil puisse être restitué par le premier, c'est-à-dire $A \cdot B = I$. On voit que le système ne pourra fonctionner que si l'une des machines est "câblée" avec la matrice A et l'autre, avec la matrice A^{-1}. Les 2 correspondants pourront changer de code aussi souvent qu'ils le voudront. Il leur suffira d'utiliser chaque fois une paire de matrices (A, A^{-1}).

3.2.4 Méthode de calcul de A^{-1}

a) La définition même de l'inversibilité de A fournit un point de départ à notre investigation. En effet, par définition, A est inversible si et seulement si il existe une matrice carrée B, de même ordre, telle que $A \cdot B = B \cdot A = I$.

On a donc à résoudre deux équations matricielles, $AB = I$ et $B'A = I$, et on aimerait qu'elles admettent la même solution.

Soit $B = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \cdots & & & \cdots \\ x_{n1} & x_{n2} & \cdots & x_{nn} \end{bmatrix}$ et $I = \begin{bmatrix} 1 & 0 & \cdots & 0 \\ 0 & 1 & \cdots & 0 \\ \cdots & & & \cdots \\ 0 & \cdots & \cdots & 1 \end{bmatrix}$

Élaboration d'une stratégie

— Résoudre le système $A \cdot B = I$; c'est un système de n^2 équations à n^2 variables.

— Si le système ne possède pas de solution, on conclura que A n'est pas inversible.

— Que faire si le système possède une infinité de solutions?

Réfléchissons un peu. Avant même de toucher aux équations $B'A = I$ et $AB = I$, on sait que si A est inversible, alors chacune de ces équations a une solution unique et $B = B' = A^{-1}$ (c'est l'énoncé même du théorème 1 (section 3.2.2)).

Donc, si le système possédait une infinité de solutions, il faudrait en conclure que A n'est pas inversible.

— Si le système $A \cdot B = I$ possède une solution unique, celle-ci constitue une excellente candidate au "poste" d'inverse de A.

Pour en avoir le coeur net, on pourrait

• ou bien résoudre le système $B' \cdot A = I$. S'il possède une solution, elle sera nécessairement unique, puisqu'on sait déjà que $A \cdot B = I$ et que le théorème 1 impose dans ce cas l'égalité de B et B'.

• ou bien calculer $B \cdot A$ et vérifier que l'on obtient bien I.
La 2ème solution paraît, pour le moment, plus rapide que la première.

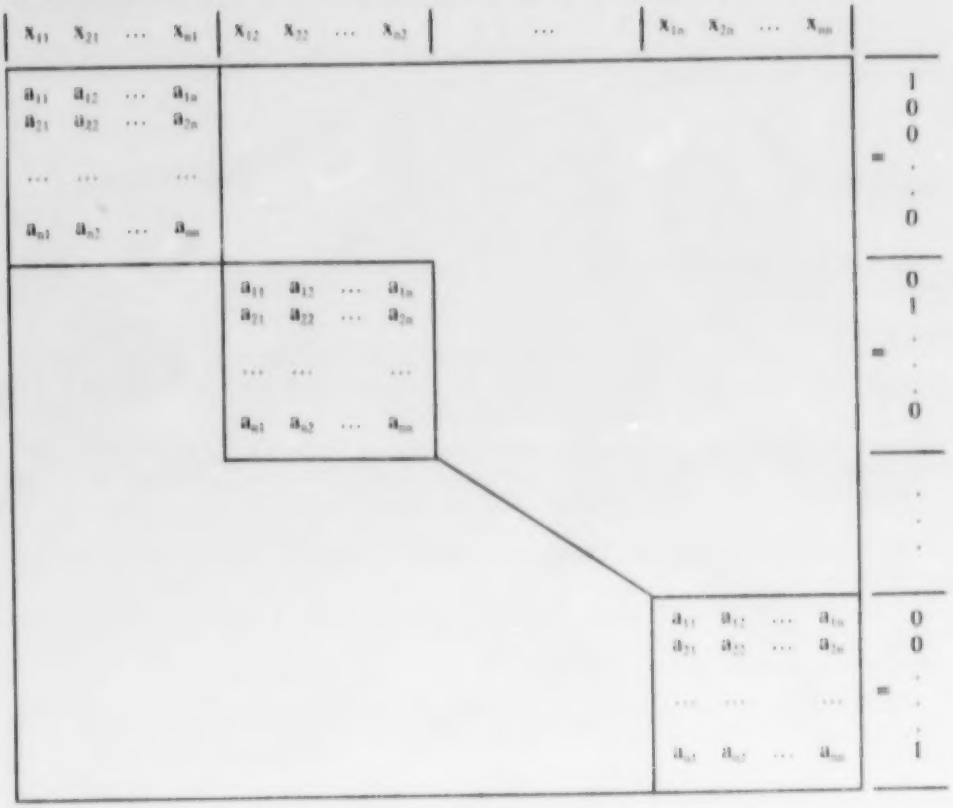
Réflexion sur cette stratégie
Il serait bon d'estimer ce qu'il en "coûterait" d'exécuter le plan d'action qu'on vient de bâtir. C'est le but de l'exercice qui suit.

Exercice
(i) Évaluer la complexité de la résolution du système $A \cdot B = I$, en se servant des résultats obtenus à la section 3.1.8.
(ii) Évaluer le nombre maximum d'additions et de multiplications requises pour effectuer le produit de deux matrices carrées d'ordre n (revoir la définition de cette opération).

Réponse
(i) Il faut compter de l'ordre de $\frac{n^3}{3}$ additions et multiplications; dans le cas d'une (petite) matrice carrée d'ordre 10, cela représente 300 000 additions et multiplications.
(ii) Environ n^3 additions et multiplications.

La stratégie élaborée — bien que théoriquement correcte — apparaît comme extrêmement coûteuse. Tout effort d'amélioration devrait porter en priorité sur la résolution de $A \cdot B = I$ parce que c'est cette partie qui requiert — et de loin — le plus grand nombre d'opérations. On va examiner ce système de plus près dans le but de simplifier les calculs, si possible.

Analyse du système $A \cdot B = I$
Ce système d'équations a une structure remarquable: il se décompose en blocs.



Ce système est bien moins terrible qu'il n'y paraît à première vue. Si on résout chaque sous-système séparément, cela représente n systèmes de n équations à n variables. Pour de grandes valeurs de n, ceci requiert de l'ordre de $n \cdot \frac{n^3}{3}$ additions et multiplications ($\frac{n^4}{3}$ vaut environ 3000, pour $n = 10$). C'est bien plus raisonnable que $\frac{n^6}{3}$.

Mais il y a mieux!
Que constate-t-on quand on compare ces n systèmes?
On retrouve la même matrice, soit A, dans chaque bloc.

Comme la méthode de Gauss n'opère que sur les lignes de la matrice A et que la nature et l'ordre des opérations d'élimination ne dépendent aucunement du membre de droite, on aboutit à la même forme échelonnée-réduite de A dans chacun des n systèmes. Seul le membre de droite diffère d'un système à un autre. Il suffit donc d'opérer une seule fois sur la matrice A; chaque opération de réduction de la matrice A sera appliquée individuellement au membre de droite de chaque système.

En résumé, on applique les opérations de réduction à la matrice augmentée $[\, A \mid I \,]$. Chaque opération effectuée sur une ligne de A est répétée pour la même ligne de I.

Supposons que le système ne possède qu'une seule solution.

Il est alors facile de voir que A est équivalente-ligne à I. Soit $[\, I \mid Z \,]$ le système réduit obtenu par la méthode de Gauss. Désignons par Z^1, la 1ère colonne de Z, par Z^2, sa 2ème colonne, ... par Z^n, sa $n^{ème}$ colonne.

Le 1er sous-système de $A \cdot B = I$ donne:

$$\begin{bmatrix} x_{11} \\ x_{21} \\ \dots \\ x_{n1} \end{bmatrix} = Z^1. \text{ Le 2ème sous-système donne } \begin{bmatrix} x_{12} \\ x_{22} \\ \dots \\ x_{n2} \end{bmatrix} = Z^2. \text{ le } n^{ème} \text{ donne } \begin{bmatrix} x_{1n} \\ x_{2n} \\ \dots \\ x_{nn} \end{bmatrix} = Z^n.$$

On a donc $B = Z$.

L'hypothèse " $[\, A \mid I \,]$ possède une infinité de solutions" sera examinée ultérieurement.

Exercice
Calculer la complexité de l'opération qui consiste à ramener $[\, A \mid I \,]$ à la forme échelonnée-réduite, quand A est inversible.

Réponse:
Pour de grandes valeurs de n, elle est de l'ordre de $2 \cdot \frac{n^3}{3}$ additions et multiplications. Pour $n = 10$, cela va chercher dans les 600 additions et multiplications.

Commentaires
(i) On est passé de $\frac{n^6}{3}$ opérations à $\frac{n^4}{3}$ opérations, puis à $\frac{2n^3}{3}$.

(ii) La résolution de $A \cdot B = I$ puis le produit $B \cdot A$ requièrent de l'ordre de

$\frac{2n^3}{3} + n^2 = \frac{5n^3}{3}$ additions et multiplications.

(iii) Il est temps maintenant de mettre l'accent sur la réduction du second terme, soit n^3.

Comment multiplier deux matrices carrées d'ordre n en un nombre d'opérations substantiellement plus petit que n^3? Les informaticiens ont mis au point des techniques de produit matriciel qui représentent un gain appréciaIbe par rapport à n^3. Mais ne pourrait-on pas être dispensé du calcul du produit $B \cdot A$?

Bon. On sait que le produit matriciel n'est pas commutatif. S'il l'était, la question serait réglée, puisque $A \cdot B$ est déjà connu. Il existe bien des matrices qui commutent, c'est-à-dire, des matrices X et Y qui sont telles que $XY = YX$. (On en a vu quelques exemples en classe). Si on pouvait **caractériser** les matrices A et B pour lesquelles $BA = AB$ dès que $AB = I$, on n'aurait pas à se casser la tête à calculer le produit $B \cdot A$ pour ces matrices. D'où le problème suivant:

Problème
Caractériser les matrices A et B pour lesquelles $AB = I$ entraîne $BA = I$.

Analyse du problème
Toutes les matrices inversibles répondent à la question. Mais cela ne nous est d'aucun secours, parce que, au moment où on trouve B telle que $A \cdot B = I$, on ne sait pas si A est inversible. On commence à suspecter que $A \cdot B = I$ suffit pour établir l'inversibilité de A. Cela nous amène à revoir la formulation du problème.

Nouveau problème
Est-il vrai que $AB = I \Longrightarrow BA = I$, pour **toute** paire de matrices (A, B)?

Analyse
On veut savoir si $AB = I$ entraîne que
(i) A et B sont inversibles et
(ii) B est l'inverse de A.

Lemme Ce résultat est vrai si $AB = I$ ne possède qu'une seule solution.

Preuve
Sous l'hypothèse que $[\, A \mid I \,]$ ne possède qu'une seule solution, c'est-à-dire que A est équivalente-ligne à I, on peut procéder ainsi: il existe une suite $O_1, O_2, \ldots, O_k$ de k opérations élémentaires, telle que

$$O_k (O_{k-1} (\ldots (O_2 (O_1 (A))) \ldots) = I$$

ou encore, il existe k matrices élémentaires $E_1, E_2, \ldots, E_k$, telles que $E_k \cdot E_{k-1} \ldots E_2 \cdot E_1 \cdot A = I$. Les matrices élémentaires sont inversibles et on obtient $A = E_1^{-1} E_2^{-1} \ldots E_k^{-1}$. A est inversible d'après le corollaire 2 (section 3.2.2). A inversible et $AB = I$ entraînent $B = A^{-1}$. □

On peut en déduire l'"algorithme" suivant:

1ère Étape
Résoudre $[\, A \mid I \,]$.

2ème Étape
Si $[\, A \mid I \,]$ ne possède aucune solution, A n'est pas inversible.
Si $[\, A \mid I \,]$ possède une infinité de solutions, A n'est pas inversible.
Si $[\, A \mid I \,]$ possède une seule solution, ce système se réduit à $[\, I \mid A^{-1} \,]$.

Cet "algorithme" — irréprochable en pratique — contient cependant une incorrection: $AB = I$ ne peut pas posséder une infinité de solutions, comme on va le voir.
On va prouver que si $A \cdot B$ est inversible, alors A et B le sont.
Pour cela, on a besoin d'un résultat préliminaire.

Théorème 3
A est inversible si et seulement si $AX = 0$ possède une solution unique.

Preuve
(i) A est inversible $\Longrightarrow AX = 0$ possède une seule solution.
Puisque A est inversible, A^{-1} existe, et en multipliant à gauche AX et 0 par A^{-1} on trouve $A^{-1}(AX) = A^{-1}0$, soit $X = 0$. Donc $X = 0$ est l'unique solution de cette équation.
(ii) $AX = 0$ possède une seule solution $\Longrightarrow$ A est inversible.
Si $AX = 0$ possède une seule solution, A est équivalente-ligne à I et donc — d'après la preuve du lemme précédent — A est inversible. □

Théorème 4
Si $A \cdot B$ est inversible, il en est de même de A et B.

Ce théorème admet le corollaire suivant.

Corollaire 3
Si $A \cdot B = I$, alors A est inversible et B est l'inverse de A.

Preuve du théorème 4
La contraposée de cet énoncé se lit comme suit:
Si A ou B n'est pas inversible, alors $A \cdot B$ ne l'est pas.
• Si B n'est pas inversible, il existe une infinité de vecteurs X de dimension n pour lesquels $BX = 0$. Alors, pour ces vecteurs, on a aussi $A(BX) = A \cdot 0$, ou $(AB)X = 0$.
• Supposons B inversible et A non inversible. L'équation $A \cdot Y = 0$ possède une infinité de solutions. Pour chacune de ces solutions $\bar{Y}$, l'équation $B \cdot X = \bar{Y}$ possède une solution $\bar{X}$, parce que B est inversible. Il existe donc une infinité de vecteurs $\bar{X}$ tels que $(AB)\bar{X} = A(B\bar{X}) = A\bar{Y} = 0$. □

Exercice
(i) Comparer l'énoncé du théorème 4 à celui du théorème 2.
(ii) Expliquer pourquoi l'"algorithme" défini plus haut est "irréprochable en pratique".

On conclut avec un algorithme.

ALGORITHME (Calcul de la matrice inverse de A)

1ère Étape
Résoudre $[A \mid I]$ en mettant A sous forme échelonnée-réduite.

2ème Étape
Si le système ne possède aucune solution, A n'est pas inversible.
Sinon, on obtient $[I \mid A^{-1}]$.

Fin

Complexité: de l'ordre de $\frac{2n^3}{3}$.

Remarque
Cette façon de présenter et de justifier l'algorithme n'est sans doute pas la plus courte et on pourrait lui préférer celle de [2], par exemple.

On doit toutefois se garder contre la tentation d'en escamoter une partie comme le font certains auteurs. On résout l'équation $A \cdot B = I$ mais on ne souffle mot de l'équation $B \cdot A = I$. Or, s'il est vrai que $A \cdot B = I$ entraine $BA = I$, cela est loin d'être évident, comme en témoignent les jongleries qui précèdent.

Exercice (pour ceux qui connaissent la méthode de la matrice adjointe)
Calculer la complexité de la méthode de la matrice adjointe et la comparer à $\frac{2n!}{3}$.

3.3 Trouver une Famille de Vecteurs Linéairement Indépendants, de Taille Maximale

Référence: [7]

3.3.1 Énoncé du problème

Soit E, un espace vectoriel. Soit S, une famille finie de vecteurs de E. Trouver une sous-famille S' de S, linéairement indépendante et de cardinalité maximale.

3.3.2 Utilisation

Cours de Math 105, Math 205

3.3.3 Connaissances requises

a) Méthode d'élimination de Gauss.
b) Espace vectoriel; systèmes de générateurs; systèmes de vecteurs linéairement indépendants; bases
c) **Théorème 1**
Toutes les bases d'un espace vectoriel ont le même nombre de vecteurs.
Définition 1
La dimension d'un espace vectoriel est égale au nombre de vecteurs d'une base quelconque de cet espace.
Théorème 2
Soit E, un espace vectoriel de dimension n. Tout système linéairement indépendant de vecteurs de E possède au plus n vecteurs.
d) **Théorème 3**
Soit $S = \{ v_1 , v_2 , \ldots , v_r \}$, un ensemble de r vecteurs linéairement indépendants d'un espace vectoriel E. Soit F, le sous-espace vectoriel de E engendré par S. Soit v, un vecteur quelconque de E. v n'appartient pas à l'espace F si et seulement si $T = S \cup \{v\}$ est linéairement indépendant.

Preuve
(i) $v \in F \Longrightarrow$ T n'est pas linéairement indépendant.
En effet, si $v \in F$, alors il existe r nombres $k_1 , k_2 , \ldots , k_r$, tels que
$v = k_1v_1 + k_2v_2 + \ldots + k_rv_r$.
Il s'ensuit que $1 \cdot v - k_1v_1 - k_2v_2 - \ldots - k_rv_r = \vec{0}$

(ii) T n'est pas linéairement indépendant $\Longrightarrow v \in F$
Si T n'est pas linéairement indépendant, il existe $k , k_1 , k_2 , \ldots , k_r$ **non tous nuls**, tels que $kv + k_1v_1 + \ldots + k_rv_r = \vec{0}$. Si k est égal à zéro, on a $k_1v_1 + \ldots + k_rv_r = \vec{0}$ avec au moins un des coefficients $k_1 , \ldots , k_r$ non nul; ceci contredit le fait que S est linéairement indépendant. Donc $k \neq 0$ et $v = \left(-\frac{k_1}{k}\right)v_1 + \ldots + \left(-\frac{k_r}{k}\right)v_r$

3.3.4 Application

Soit E un espace vectoriel dont on connaît un système de générateurs.
(i) Trouver une base de E
(ii) Compléter un système linéairement indépendant de vecteurs de E de façon à former une base.

3.3.5 Ébauche d'une stratégie

Le théorème 3 (section 3.3.3) suggère de partir avec un vecteur v_i de S, non nul (non nul parce qu'on veut qu'il soit linéairement indépendant).

a) Si chacun des autres vecteurs de S est une combinaison linéaire de v_i, on conclut que $S' = \{ v_i \}$ répond à la question.
b) S'il existe un vecteur v_j de S tel que $\{ v_i , v_j \}$ est linéairement indépendant, on réitère le processus indiqué en a): si chacun des autres vecteurs de S est une combinaison linéaire de v_i et v_j, on conclut que $\{ v_i , v_j \}$ répond à la question, etc...

3.3.6 Analyse

La stratégie énoncée ne dit pas grand chose du choix des vecteurs: si tous les vecteurs de S sont non nuls, n'importe lequel pourrait être choisi pour jouer le rôle de v_i; de même s'il existe w et w' tels que $\{ v_i , w \}$ et $\{ v_i , w' \}$ sont linéairement indépendants, on peut choisir $v_j = w$ ou $v_j = w'$. Il est donc légitime de se demander:

— Que faire quand on a le choix entre plusieurs vecteurs? Faut-il opter pour le premier qui nous tombe sous la main ou y a-t-il quelque critère pour un "meilleur" choix? Quelle influence le choix des vecteurs a-t-il sur la cardinalité de S"?

Si on examine la définition 1 (section 3.3.3), elle ne dit rien non plus du choix des vecteurs. Elle semble indiquer que — peu importe la manière dont un ensemble de vecteurs a été assemblé — s'il forme une base de E, il doit avoir autant de vecteurs que n'importe quelle autre base. Peut-être que la même règle prévaut ici aussi.

Théorème 4
La stratégie énoncée à la section 5 fournit une solution au problème posé, quelque soit le vecteur choisi à chaque itération.

Preuve
Commençons par exprimer cette stratégie sous la forme d'un algorithme.

Algorithme (Construction d'une famille de vecteurs linéairement indépendants de cardinalité maximale.)

Étape 1
Soit $S = (w_1, w_2, \ldots , w_n)$. (Les éléments de S sont ordonnés de façon tout-à-fait arbitraire et sont examinés séquentiellement. Soit $S'_1 = \{ v_1 \}$, où v_1 est le premier vecteur non nul de S.

Étape 2
Supposons $S'_i = (v_1 , v_2 , \ldots , v_i)$ pour une certaine valeur de i ($i \geq 1$). Supposons que $v_i = w_j$. i.e. le i^{e} élément choisi est égal au j^{e} élément de S.

— ou bien il existe un élément w de S tel que $S'_i \cup \{ w \}$ est linéairement indépendant. Soit w_k , le premier élément de S pour lequel cela a lieu. On a nécessairement $k > j$.
— ou bien tout élément de S est une combinaison linéaire des vecteurs de S'_i .
Dans le 1er cas, on pose $v_{i+1} = w_k$ et $S'_{i+1} = (v_1 , \ldots , v_{i+1})$, et on retourne au début de l'Étape 2, avec i remplacé par i + 1.

Dans le 2ème cas, on pose $S' = S'_i$ et on s'arrête.

Fin.

À chaque itération de l'algorithme, S'_i engendre un certain sous-espace vectoriel F_i de E. Comme S'_i est linéairement indépendant, il est en fait une base de F_i. Toutes les bases de F_i ont le même nombre de vecteurs, soit i (théorème 1) et tout système linéairement indépendant de F_i possède au plus i vecteurs (théorème 2). Cette propriété se conserve à chaque itération et à la fin, S' est une base de F, l'espace vectoriel engendré par S. □

Remarque À l'étape 2, on teste si $S'_i \cup \{ w_k \}$ est linéairement indépendant, **seulement** pour les valeurs de k qui sont **supérieures** à j. On essaie d'abord w_{j+1}, puis w_{j+2}, ... On s'arrête dès qu'on en a trouvé un pour lequel cela marche. Autrement, on épuise la liste des candidats.

3.3.7 Comment rendre cet algorithme opérationnel?

Étant donné S'_i et un vecteur w, comment tester si $S'_i \cup \{ w \}$ est linéairement indépendant? On peut chaque fois, se référer à la définition de l'indépendance linéaire et partir de zéro. Quand on connaît les coordonnées des vecteurs de S dans une certaine base de E, on peut utiliser le principe de l'élimination de Gauss pour réduire les calculs le plus possible.

On peut considérer les vecteurs de S comme des n-uplets, où n est la dimension de E. Quand v_i est choisi, on définit la matrice

$$A_i = [\, a_{11}, a_{12}, \ldots, a_{1n} \,]$$

avec les coordonnées de v_1.
Pour savoir si $\{ v_1, w \}$ est linéairement indépendant, on définit la matrice

$$A = \begin{bmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ b_1 & b_2 & \ldots & b_n \end{bmatrix}$$

où $b_1, b_2, \ldots, b_n$ sont les coordonnées de w. Puis on la réduit à la forme échelonnée. Si la deuxième ligne ne contient que des 0, on conclut que $\{ v_1, w \}$ n'est pas linéairement indépendant. Sinon, on a une matrice A_2 sous forme échelonnée.

Dans le cas général, on a une matrice A_i de dimension i x n, écrite sous forme échelonnée. On lui ajoute une $(i+1)^{e}$ ligne, faite des coordonnées d'un vecteur w, soit A_* la matrice ainsi obtenue. On réduit A_* à la forme échelonnée. Si la dernière ligne ne contient que des zéros, on passe au vecteur suivant. Sinon, on vient de trouver S'_{i+1} et la matrice A_{i+1} qui lui correspond, écrite sous forme échelonnée.

3.4 Programmation Linéaire; Algorithme du Simplexe

Référence: [10]

3.4.1 Utilisation

Cours de Math. 205

3.4.2 Connaissances Requises

Les règles élémentaires de l'algèbre.

L'algorithme du simplexe peut être exposé d'une façon qui la rende accessible à des étudiants ayant un minimum de connaissances en mathématiques.

3.4.3 Mise en situation

Les textes de programmation linéaire regorgent de problèmes concrets simples (composition de menus, problèmes de transport ou d'affectation, profits à maximiser sous contraintes, ...) qu'il est facile de modéliser sous la forme d'un programme linéaire ([14], [52])

On peut donc commencer par présenter trois ou quatre exemples simples dont la résolution se ramène à

maximiser $c_1x_1 + c_2x_2 + \ldots + c_nx_n$, sachant que les variables $x_1, \ldots, x_n$ sont soumises aux contraintes suivantes:

$$\begin{aligned} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n &\leq b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n &\leq b_2 \\ \cdots\cdots\cdots\cdots\cdots\cdots\cdots\cdots & \\ a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n &\leq b_m \end{aligned}$$

$$x_1 \geq 0 ; x_2 \geq 0 ; \ldots ; x_n \geq 0$$

On choisira des exemples qui s'écrivent "directement" sous cette forme ("standard"), sans qu'on ait besoin, par exemple, de renverser le sens des inégalités ou de changer le signe de la fonction à optimiser. Quand les étudiants auront compris le fonctionnement de la méthode du simplexe, on pourra revenir à la modélisation de problèmes de programmation linéaire et montrer comment il est possible de toujours ramener ces problèmes à la forme standard définie plus haut.

3.4.4 Définitions et Exemples

Considérons le programme linéaire suivant.

maximiser $3x_1 + x_2$
sous les contraintes

$$(1) \qquad \begin{aligned} 3x_1 - x_2 &\leq 3 \\ 2x_1 + x_2 &\leq 7 \\ -x_1 + 2x_2 &\leq 9 \end{aligned}$$

$$x_1 \geq 0 ; x_2 \geq 0$$

Il y a ici **2 variables**, x_1 et x_2. La **fonction à maximiser** est **linéaire** en x_1 et x_2. Les **contraintes** sont des **inéquations linéaires** en x_1 et x_2. Elles sont au nombre de cinq; les deux dernières contraintes ont une forme spéciale: ce sont des **contraintes de non-négativité**. La fonction linéaire à maximiser s'appelle **la fonction objective** du problème.

Si on donne à x_1 la valeur $\frac{1}{2}$ et à x_2 la valeur 3, on définit une **solution réalisable** du problème: toutes les contraintes sont satisfaites par ces deux valeurs. La fonction objective prend alors la valeur $\frac{15}{2}$. Résoudre ce programme linéaire consiste à trouver au moins une valeur de x_1 et de x_2 pour laquelle toutes les contraintes sont satisfaites et la fonction objective prend la plus grande valeur possible. **La seule solution** à ce problème consiste à attribuer la valeur 2 à x_1 et la valeur 3 à x_2. La fonction objective prend alors la valeur 9. Il est possible d'obtenir pour la fonction objective des valeurs supérieures à 9 (en fixant par exemple x_1 à 3 et x_2 à 1) mais alors, les contraintes ne seront pas toutes satisfaites. Une solution réalisable pour la-

quelle la fonction objective prend la plus grande valeur possible est une **solution optimale** du problème. La fonction objective atteint sa **valeur optimale** pour toute solution optimale.

Le programme linéaire (1) possède une solution optimale et une seule. Il n'en est pas toujours ainsi.

• Il existe des programmes linéaires qui possèdent **une infinité de solutions optimales** (la valeur optimale, elle, est unique). Ainsi, le programme linéaire

$$
\begin{aligned}
&\text{maximiser } 5x_1 - \tfrac{1}{2}x_2 + \tfrac{1}{2}x_3 \\
&\text{sous les contraintes} \\
&\quad x_1 + x_2 - x_3 \leqslant 1 \\
&\quad x_1 - x_2 + x_3 \leqslant 1 \\
&\quad -x_1 + x_2 + x_3 \leqslant 1 \\
&\quad x_1 \geqslant 0 \\
&\quad x_2 \geqslant 0 \\
&\quad x_3 \geqslant 0
\end{aligned}
\qquad (2)
$$

possède une infinité de solutions: tous les triplets (x_1, x_2, x_3) pour lesquels $x_1 = 1$; $x_2 = x_3$ et $0 \leqslant x_2 \leqslant 1$. La valeur optimale est 5.

• Certains programmes linéaires ne possèdent aucune solution réalisable. Par exemple,

$$
\begin{aligned}
&\text{maximiser } x_1 - 2x_2 \\
&\text{sous les contraintes} \\
&\quad 2x_1 + x_2 \leqslant 7 \\
&\quad -6x_1 - 3x_2 \leqslant -23 \\
&\quad x_1 \geqslant 0 \\
&\quad x_2 \geqslant 0
\end{aligned}
\qquad (3)
$$

• Enfin, certains programmes linéaires, bien qu'ayant des solutions réalisables, ne possèdent aucune solution optimale. Ils ne possèdent pas non plus de valeur optimale. On dit que ces problèmes sont **non bornés**. Un exemple d'un tel programme linéaire est donné par

$$
\begin{aligned}
&\text{maximiser } x_1 + x_2 \\
&\text{sous les contraintes} \\
&\quad 3x_1 - x_2 \leqslant 3 \\
&\quad -x_1 - 2x_2 \leqslant 0 \\
&\quad x_1 \geqslant 0 \\
&\quad x_2 \geqslant 0
\end{aligned}
\qquad (4)
$$

Par exemple, tous les couples (x_1, x_2) où $x_1 = 1$ et $x_2 \geqslant 0$ définissent des solutions réalisables. Pour une telle solution, la fonction objective prend la valeur $1 + x_2$. Cette valeur tend vers l'infini avec x_2.

Tout programme linéaire rentre dans l'une des catégories que l'on vient de définir: il possède au moins une solution optimale, il ne possède aucune solution réalisable ou il est non borné.

3.4.5 Histoire de la programmation linéaire

voir [10] et [14] .

3.4.6 La méthode du simplexe: comment cela marche

(L'exposé de Chvátal ([10] , chapitre 2) sert de modèle à la présente section)

Soit à résoudre le programme linéaire

$$
\begin{aligned}
&\text{maximiser } z = 3x_1 + x_2 \\
&\text{sous les contraintes} \\
&\quad 3x_1 - x_2 \leqslant 3 \\
&\quad 2x_1 + x_2 \leqslant 7 \\
&\quad -x_1 + 2x_2 \leqslant 9 \\
&\quad x_1 \geqslant 0 ; x_2 \geqslant 0
\end{aligned} \tag{5}
$$

Stratégie

i) Déterminer une solution réalisable — s'il en existe une. (On verra plus loin comment faire pour y parvenir. Pour le moment, il suffit de supposer connue une telle solution.)

ii) Soit $x = (x_1, x_2, \ldots, x_n)$, une solution réalisable connue, et z, la valeur correspondante de la fonction objective. Déterminer (on ne sait pas encore comment) une autre solution réalisable $\bar{x} = (\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n)$ de telle sorte que $\bar{z}$, la valeur correspondante de la fonction objective, soit supérieure à z.

iii) Répéter ii) jusqu'au moment où plus aucune amélioration n'est possible, i.e. jusqu'au moment où z ne peut plus augmenter. (Cela suppose la définition d'un **critère d'optimalité**).

Cette stratégie semble assez naturelle, mais elle soulève plus de problèmes qu'elle n'en résout. Essayons néanmoins de la préciser à l'aide de (5).

a) Il est facile de trouver une solution réalisable pour (5), par exemple $x^0 = (0, 0)$ et $z^0 = 0$.

b) Les contraintes de non-négativité sont extrêmement simples et il est immédiat de vérifier si elles sont satisfaites par une solution donnée. Les autres contraintes sont bien moins pratiques et il serait plus commode de travailler avec des équations plutôt qu'avec des inégalités.

L'inégalité $3x_1 - x_2 \leqslant 3$ s'interprète ainsi: l'écart entre 3 et la valeur de $3x_1 - x_2$ est non négatif.

Donc si on définit cet écart par $3 - (3x_1 - x_2)$, on a: $3x_1 - x_2 \leqslant 3$ est équivalent à

$$\begin{cases} x_3 = 3 - 3x_1 + x_2 & \text{et} \\ x_3 \geqslant 0 \end{cases}$$

De même, $2x_1 + x_2 \leqslant 7$ est équivalent à $\begin{cases} x_4 = 7 - 2x_1 - x_2 \\ x_4 \geqslant 0 \end{cases}$

Finalement, $-x_1 + 2x_2 \leqslant 9$

est équivalent à $\begin{cases} x_5 = 9 + x_1 - 2x_2 & \text{et} \\ x_5 \geqslant 0 \end{cases}$

Donc, le programme linéaire (5) est équivalent à

$$
\begin{aligned}
&\text{maximiser } z = 3x_1 + x_2 \\
&\text{sous les contraintes} \\
&\quad x_3 = 3 - 3x_1 + x_2 \\
&\quad x_4 = 7 - 2x_1 - x_2 \\
&\quad x_5 = 9 + x_1 - 2x_2 \\
&\quad x_1 \geqslant 0 ; x_2 \geqslant 0 ; x_3 \geqslant 0 \\
&\quad x_4 \geqslant 0 ; x_5 \geqslant 0
\end{aligned} \tag{6}
$$

Étant donné une solution réalisable (x_1, x_2) de (5), il suffit de calculer x_3, x_4 et x_5 selon la façon dont elles sont définies dans (6). x_3, x_4, x_5 seront nécessairement non-négatives puisque (x_1, x_2) est une solution **réalisable** de (5). Inversement, si $(x_1, x_2, x_3, x_4, x_5)$ est une solution réalisable de (6), alors (x_1, x_2) est une solution réalisable de (5).

Les variables x_1 et x_2 sont les **variables de décision** du problème, tandis que x_3, x_4 et x_5 en sont les **variables d'écart.**

Une solution réalisable de (6) est $x^0 = (x_1^0, \ldots, x_9^0) = (0, 0, 3, 7, 9)$. La valeur qui lui correspond est $z^0 = 0$

c) On est à la recherche d'une solution réalisable $x^1 = (x_1^1, x_2^1, \ldots, x_9^1)$ telle que $z^1 > z^0$. Pour cela il faut augmenter x_1 ou x_2. Dans notre désir d'accroître z^0 le plus possible et le plus rapidement possible, nous pourrions envisager d'augmenter x_1 et x_2 en même temps. Mais l'expérience enseigne qu'il vaut mieux modifier les conditions d'un problème une à la fois.
Augmentons x_2 par exemple en gardant x_1 à zéro.
On a les résultats suivants:

$$\begin{array}{l} x_1 = 0\,;\ x_2 = 1\,;\ x_3 = 4\,;\ x_4 = 6\,;\ x_5 = \ \ 7\,;\ z = 1 \\ x_1 = 0\,;\ x_2 = 2\,;\ x_3 = 5\,;\ x_4 = 5\,;\ x_5 = \ \ 5\,;\ z = 2 \\ x_1 = 0\,;\ x_2 = 3\,;\ x_3 = 6\,;\ x_4 = 4\,;\ x_5 = \ \ 3\,;\ z = 3 \\ x_1 = 0\,;\ x_2 = 4\,;\ x_3 = 7\,;\ x_4 = 3\,;\ x_5 = \ \ 1\,;\ z = 4 \\ x_1 = 0\,;\ x_2 = 5\,;\ x_3 = 8\,;\ x_4 = 2\,;\ x_5 = -1\,;\ z = 5 \end{array}$$

Les quatre premières solutions sont réalisables; la cinquième ne l'est pas parce que x_5 est négative. Donc quand on augmente x_2 tout en gardant x_1 à zéro, il faut s'arrêter quelque part entre la valeur 4 et la valeur 5. Mais où exactement faut-il s'arrêter?

Les contraintes $\begin{cases} x_3 = 3 - 3x_1 + x_2 \\ x_1 = 0 \\ x_3 \geqslant 0 \end{cases}$ nous indiquent que x_2 doit être supérieure ou égale à -3, ce qui n'est nullement gênant puisque x_2 prend des valeurs non-négatives.

les contraintes $\begin{cases} x_4 = 7 - 2x_1 - x_2 \\ x_1 = 0 \\ x_4 \geqslant 0 \end{cases}$ sont satisfaites quand x_2 est inférieure ou égale à 7. Finalement, les contraintes $\begin{cases} x_5 = 9 + x_1 - 2x_2 \\ x_1 = 0 \\ x_5 \geqslant 0 \end{cases}$ forcent x_2 à être inférieure ou égale à $\frac{9}{2}$. Toutes les contraintes sont satisfaites par $x_1 = 0$, $x_2 = \frac{9}{2}$. C'est le plus loin qu'on puisse aller dans la direction définie par $x_1 = 0$, $x_2 \geqslant 0$.
Cela donne:

$$x^1 = (0, \tfrac{9}{2}, \tfrac{15}{2}, \tfrac{5}{2}, 0)\,;\ z^1 = \tfrac{9}{2}$$

d) (i) **Exercice**
Répéter le processus précédent en gardant la valeur de x_2 fixée à $\frac{9}{2}$ et en augmentant x_1.
Quelle est la plus grande valeur que peut prendre x_1?
Calculer alors x_3, x_4, x_5 et z.

Réponse: $(\frac{5}{4}, \frac{9}{2}, \frac{15}{4}, 0, \frac{5}{4})$ et $z = \frac{23}{4}$

(ii) Maintenant,
— si on augmente x_2 en gardant x_1 fixée à $\frac{5}{4}$, x_4 devient négative. On ne peut donc pas répéter le même processus à nouveau.
— Serait-ce parce que l'on a atteint une solution optimale? Non. On peut encore augmenter z en augmentant "légèrement" x_1 et en diminuant x_2.

Ainsi,
Si x_1 devient $\frac{5}{4} + \frac{1}{2} = \frac{7}{4}$ et
si x_2 devient $\frac{9}{2} - 2 \cdot \frac{1}{2} = \frac{7}{2}$, alors
x_3, x_4, x_5 et z sont changées en $\frac{5}{2}$, 0, $\frac{15}{4}$ et $\frac{23}{4}$, respectivement.

— On est donc dans l'incapacité d'obtenir une meilleure solution en faisant varier une seule des deux variables, x_1 et x_2. Nous allons essayer de contourner cette difficulté.

c) x^0 et z^0 correspondent au programme linéaire (6). x_1^0 et x_2^0 sont fixées à zéro et dans l'expression (6), les autres variables ainsi que z s'expriment en fonction de x_1 et x_2.
Dans x^0, x_1^0 et x_2^0 prennent la valeur 0.

Déterminons une expression équivalente à (6) dans laquelle x_2, x_3, x_4 et z seront écrites en fonction de x_1 et x_5.

De $x_3 = 9 + x_1 - 2x_2$ nous déduisons $x_2 = \frac{1}{2}(9 + x_1 - x_3)$

En remplaçant x_2 par sa valeur dans l'expression de x_3, x_4 et z dans (6), on obtient

$$
\begin{aligned}
&\text{maximiser } z = \tfrac{3}{2}x_1 - \tfrac{1}{2}x_3 + \tfrac{7}{2} \\
&\text{sous les contraintes} \\
&x_2 = \tfrac{9}{2} + \tfrac{1}{2}x_1 - \tfrac{1}{2}x_3 \\
&x_3 = \tfrac{15}{2} - \tfrac{3}{2}x_1 - \tfrac{1}{2}x_5 \\
&x_4 = \tfrac{1}{2} - \tfrac{1}{2}x_1 + \tfrac{1}{2}x_5 \\
&x_1 \geq 0\,;\ x_2 \geq 0\,;\ x_3 \geq 0 \\
&x_4 \geq 4\,;\ x_5 \geq 0
\end{aligned}
\qquad (7)
$$

f) Les programmes linéaires (6) et (7) sont équivalents à (5), en ce sens que ces trois programmes admettent le même ensemble-solution. Ils contiennent la même information, mais présentée différemment. Dans (5), toutes les variables sont sur un pied d'égalité; la forme (6) [resp. (7)] suggère que les variables x_1 et x_2 [resp. x_1 et x_5] sont "indépendantes", tandis que les variables x_3, x_4, x_5 [resp. x_2, x_3, x_4], ainsi que z, dépendent de x_1 et x_2 [resp. x_1 et x_5]. De plus, quand on fixe les variables "indépendantes" à 0 dans (6) ou dans (7), on trouve une solution réalisable.

On résume toutes ces propriétés de (6) et (7) en disant que ce sont des **dictionnaires réalisables.** Chaque dictionnaire réalisable décrit une solution réalisable. Celle-ci est obtenue en fixant les variables "indépendantes" à 0. Ainsi, (6) correspond à la solution (0, 0, 3, 7, 9), tandis que (7) correspond à $(0, \frac{9}{2}, \frac{15}{2}, \frac{1}{2}, 0)$.

Il existe toutefois des solutions réalisables qui ne sont décrites par aucun dictionnaire réalisable; par exemple, aucun dictionnaire ne décrit la solution $(\frac{5}{2}, \frac{9}{2}, \frac{15}{2}, 0, \frac{5}{2})$. Les solutions réalisables qui peuvent être décrites par des dictionnaires sont dites **solutions de base.** Dans une telle solution, on distingue les variables "indépendantes", appelées **variables hors-base** et les variables "dépendantes", appelées **variables de base.**

Revoyons ce qui a été fait jusqu'ici.
On est parti avec un dictionnaire réalisable (6), et la solution de base décrite par (6), soit $x^0 = (0, 0, 3, 7, 9)$, $z^0 = 0$. Dans cette solution, x_1 et x_2 sont des variables hors-base, fixées à 0. On peut accroître z^0 en faisant croître x_1 et x_2. On pourrait agir sur x_1 et x_2 en même temps, mais on décide de faire varier une variable à la fois. En augmentant x_2 tout en gardant x_1 fixée à 0, on obtient $x^1 = (0, \frac{9}{2}, \frac{15}{2}, \frac{1}{2}, 0)$, $z^1 = \frac{7}{2}$.

Là, deux voies s'offrent à nous:
• ou bien on garde x^1 et le dictionnaire (6) (qui décrit une solution de base autre que x^1). On peut accroître z^1 en augmentant la valeur de x_1. On obtient la solution
$\bar{x} = (\frac{5}{2}, \frac{9}{2}, \frac{15}{2}, 0, \frac{5}{2})$, $\bar{z} = \frac{11}{2}$. Cette solution ne correspond à aucun dictionnaire: pour continuer à accroître z, il faut agir sur plus d'une variable. Cela ressemble fort à une impasse.

• ou bien on construit (7), le dictionnaire réalisable qui décrit x^1. L'expression de z dans (7) suggère d'augmenter la valeur de x_1.

g) Dans (7), $z = \frac{1}{2}x_1 - \frac{1}{2}x_3 + \frac{9}{2}$. La valeur de z augmente avec celle de x_1, x_3 demeurant fixée à 0. La croissance de x_1 est limitée par l'expression de x_4 à 1. On obtient $x^2 = (1, 5, 5, 0, 0)$ et $z^2 = 8$.

Quand x_1 passe de la valeur 0 à la valeur limite 1, la variable x_4 — qui impose cette limite — passe de $\frac{1}{2}$ à 0. On dit que x_1 **entre dans la base** tandis que x_4 en **sort**. On va maintenant **mettre à jour le programme linéaire** en construisant le dictionnaire qui décrit x^2 (les variables hors-base de x^2 sont x_4 et x_3). L'équation $x_4 = \frac{1}{2} - \frac{1}{2}x_1 + \frac{1}{2}x_3$ donne $x_1 = 1 - \frac{2}{1}x_4 + \frac{1}{1}x_3$. On en déduit

(8)
$$\begin{aligned} &\text{maximiser } z = 8 - \tfrac{1}{2}x_4 + \tfrac{1}{2}x_3 \\ &\text{sous les contraintes} \\ &x_1 = 1 - \tfrac{1}{2}x_4 + \tfrac{1}{2}x_3 \\ &x_2 = 5 - \tfrac{1}{2}x_4 - \tfrac{3}{2}x_3 \\ &x_5 = 5 + x_4 - x_3 \\ &x_1 \geqslant 0 ;\ x_2 \geqslant 0 ;\ x_3 \geqslant 0 \\ &x_4 \geqslant 0 ;\ x_5 \geqslant 0 \end{aligned}$$

Cette opération de mise-à-jour s'appelle aussi **pivotement**.

h) On peut accroître z en agissant sur x_3. La croissance de x_3 est limitée par x_1 à 5. x_3 entre dans la base et x_1 en sort. On trouve $x^3 = (2, 3, 0, 0, 5)$, $z^3 = 9$ et

(9)
$$\begin{aligned} &\text{maximiser } z = 9 - \tfrac{1}{2}x_1 - \tfrac{0}{2}x_4 \\ &\text{sous les contraintes} \\ &x_1 = 2 - \tfrac{1}{2}x_3 - \tfrac{1}{2}x_4 \\ &x_2 = 3 + \tfrac{3}{2}x_3 - \tfrac{1}{2}x_4 \\ &x_5 = 5 - x_3 + x_4 \\ &x_1 \geqslant 0 ;\ \ldots ;\ x_5 \geqslant 0 \end{aligned}$$

j) Si on augmente x_3 ou x_4, z diminue. Donc x^3 est une solution optimale et la valeur optimale de (5) est 9.

k) L'expression de z dans (9) suggère le critère d'optimalité suivant:
Si dans l'expression de z^i tous les coefficients des variables hors-base sont non positifs, alors x^i est une solution optimale.
Si chacune des variables hors-base a un coefficient **négatif** dans z^i, x^i est l'unique solution optimale.
Si certains de ces coefficients sont nuls, les autres étant négatifs, le problème possède en général une infinité de solutions optimales. Ainsi, soit le programme linéaire

(10)
$$\begin{aligned} &\text{maximiser } z = 6 - x_3 \\ &\text{sous les contraintes} \\ &x_1 = 3 + 2x_4 + 3x_3 \\ &x_2 = 2 - 3x_4 + x_3 \\ &x_5 = 7 - x_4 - 2x_3 \\ &x_1 \geqslant 0 ;\ x_2 \geqslant 0 ;\ x_3 \geqslant 0 ;\ x_4 \geqslant 0 ; \\ &x_5 \geqslant 0 \end{aligned}$$

(3,2 7,0, 0,) est une solution optimale de (10) et 6 en est la valeur optimale. Les variables hors-base sont x_4 et x_5. Le coefficient de x_5 dans z est négatif et celui de x_4 est nul. On peut augmenter la valeur de x_4 jusqu'à $\frac{7}{3}$. Soit $x_4 = k$, avec $0 \leq k \leq \frac{7}{3}$.

($3 + 2k$, $2 - 3k$, $7 - k$, k, 0) est une solution optimale de (10), pour toute valeur de k comprise entre 0 et $\frac{7}{3}$; l'expression de z est la même pour toutes ces solutions.

En résumé

On est parti d'une ébauche de stratégie et d'un exemple numérique devant nous permettre d'éprouver cette stratégie et d'en préciser les contours. La résolution de ce problème numérique a permis de mettre au jour les principes suivants:

a) Utiliser des variables d'écart de façon à transformer le problème initial en un problème **équivalent** n'ayant que des contraintes d'égalité et des contraintes de non-négativité.

b) À chaque étape, les variables de base ainsi que la fonction objective doivent être exprimées en fonction des variables hors-base.

c) Quand dans l'expression de z toutes les variables hors-base ont un coefficient **non-positif**, la solution présente est **optimale**. Si, à ce moment-là, toutes les variables hors-base ont un coefficient **négatif** dans z, la solution présente est **l'unique solution** optimale; autrement, il peut en exister **une infinité**.

d) Si dans l'expression de z il existe des variables hors-base dont le coefficient est positif, on en choisit une, soit x_j. **En principe**, chacune des contraintes d'égalité impose une limite à l'accroissement de x_j. La limite la plus sévère détermine **au moins une** variable x_i qui doit sortir de la base pour faire place à x_j. (Il se peut qu'il y ait plusieurs contraintes d'égalité qui imposent la même limite à l'accroissement de x_j. Dans ce cas, on en choisit une). La mise-à-jour s'effectue en exprimant x_i par rapport à x_j puis en remplaçant x_i par sa valeur dans l'expression de z et celle des autres variables de base.

On a fait pas mal de chemin, mais il reste encore des points à préciser.

(i) En principe, chacune des contraintes d'égalité impose une limite à l'accroissement de x_j, la variable qui entre dans la base. Pourtant, on a rencontré des cas où cette limite n'avait rien de contraignant. Ainsi, dans (6), les contraintes $\begin{cases} x_1 = 3 - 3x_3 + x_5 \\ x_2 = 0 \\ x_3 \geq 0 \end{cases}$ imposent à x_3 d'être supérieure ou égale à -3. Qu'arrive-t-il si toutes les limitations sont aussi peu contraignantes?

(ii) Quand dans l'expression de z, plus d'une varaible a un coefficient positif, laquelle choisir comme variable entrante?

(iii) Quand plusieurs contraintes d'égalité imposent la même limite à l'accroissement de la variable entrante, laquelle choisir comme variable sortante?

(iv) Dans (6), (7), (8), (9) et (10), chaque contrainte d'égalité contient un terme constant positif. En est-il toujours ainsi? Qu'arriverait-il si un tel terme était négatif? Qu'arriverait-il si un tel terme était nul?

(v) Tout ce qui précède suppose que le processus est déjà mis en branle par la connaissance d'une solution réalisable. Mais, comment amorcer le processus?

On va essayer de répondre à ces interrogations.

(i) Construisons un programme linéaire où rien ne s'oppose à ce qu'on accroisse indéfiniment la valeur de la variable entrante. Soit, par exemple,

maximiser $z = 25 - x_1 + x_2$
sous les contraintes

$$
\begin{aligned}
x_3 &= 5 - x_1 + 3x_2 \\
x_4 &= 3 + x_1 + 2x_2 \\
x_5 &= 7 - 2x_1 + 2x_2
\end{aligned}
\qquad (11)
$$

$x_1 \geq 0$; $x_2 \geq 0$; $x_3 \geq 0$;
$x_4 \geq 0$; $x_5 \geq 0$

(Voir aussi (4)).
Les contraintes d'égalité imposent $x_1 \geq -\frac{5}{1}$; $x_2 \geq -\frac{3}{1}$; $x_3 \geq -\frac{7}{2}$. Toutes ces contraintes sont satisfaites quand x_2 est non-négative. Rien ne s'oppose à ce que x_2 tende vers $+\infty$ et alors z tend aussi vers $+\infty$ **Le problème n'est pas borné.** On remarquera que dans chaque contrainte d'égalité, le coefficient de x_2 et le terme constant sont de même signe.

(ii) Il n'y a pas de règle absolue pour briser l'égalité entre deux variables qui sont candidates à entrer dans la base. Pour des calculs effectués à la main, il est courant de choisir la candidate dont le coefficient dans z est le plus élevé. Une autre règle (la règle de Bland) consiste à choisir celle dont l'indice est le plus faible. Soit, par exemple, le programme

maximiser $12 - x_1 + 2x_2 + 5x_3$
sous les contraintes

$$
\begin{aligned}
x_4 &= 3 + x_1 - x_2 - 2x_3 \\
x_5 &= 4 - x_1 - x_2 + 5x_3
\end{aligned}
\qquad (12)
$$

$x_1 \geq 0$; $x_2 \geq 0$; $x_3 \geq 0$;
$x_4 \geq 0$; $x_5 \geq 0$

x_2 et x_3 sont des candidates pour entrer dans la base. Si on adopte la règle du plus grand coefficient, on choisira x_3 ; si on adopte la règle de Bland, on choisira x_2.

(iii) Comme précédemment, il n'y a pas de règle absolue. La règle de Bland préconise le choix de la candidate dont l'indice est le plus faible.
Voyons cela sur un exemple.

maximiser $18 - 2x_1 + 3x_2$
sous les contraintes

$$
\begin{aligned}
x_3 &= 4 + 3x_1 - 2x_2 \\
x_4 &= 6 - x_1 - x_2 \\
x_5 &= 2 - 5x_1 - x_2 \\
x_6 &= 1 + 2x_1 + 3x_2
\end{aligned}
\qquad (13)
$$

$x_1 \geq 0$; $x_2 \geq 0$; $x_3 \geq 0$;
$x_4 \geq 0$; $x_5 \geq 0$; $x_6 \geq 0$

x_2 est la variable entrante. x_5 et x_6 limitent l'accroissement de x_2 à 2. La règle de Bland suggère de choisir x_5 comme variable sortante.

Quelque soit la règle retenue, il se passe ici quelque chose d'intéressant. Supposons par exemple que x_5 soit la variable sortante (le fait que ce soit x_5 ou x_6 n'a aucune importante) et faisons la mise-à-jour. On obtient

maximiser $z = 24 + \frac{7}{3} x_3 - \frac{1}{3} x_4$
sous les contraintes

$$
\begin{aligned}
x_1 &= 2 + \tfrac{1}{3} x_3 - \tfrac{1}{3} x_4 \\
x_2 &= 4 - \tfrac{5}{3} x_3 + \tfrac{1}{3} x_4 \\
x_5 &= 0 - \tfrac{11}{3} x_3 + \tfrac{1}{3} x_4 \\
x_6 &= 7 + \tfrac{11}{3} x_3 - \tfrac{1}{3} x_4 \\
x_1 &\geqslant 0 ; \; x_2 \geqslant 0 ; \; x_3 \geqslant 0 \\
x_4 &\geqslant 0 ; \; x_5 \geqslant 0 ; \; x_6 \geqslant 0
\end{aligned}
\qquad (14)
$$

Le terme constant de l'expression de x_1 est 0. Si on avait choisi x_1 comme variable sortante, le terme constant de x_2 serait maintenant 0. Cela nous renvoie à une des questions posées dans (iv).

(iv) Si une contrainte d'égalité contient un terme constant négatif, cela indique que la solution présente n'est pas réalisable.

En effet, si $x_1 = -2 - 3x_3 + 2x_4$, x_3 et x_4 sont des variables hors-base; leur valeur dans la solution présente est 0 et alors la valeur de x_1 dans la solution présente est -2, contredisant la contrainte $x_1 \geqslant 0$. Si on est parti d'une solution réalisable et si on applique correctement l'algorithme par la suite, cela ne peut pas arriver.

Si le terme constant d'une contrainte d'égalité est nul, alors la variable de base correspondant à cette contrainte prend la valeur 0 dans la solution actuelle. Une solution de base dont au moins une des variables de base vaut zéro est dite **dégénérée**.

La dégérescence a des effets ennuyeux, comme on peut le voir sur (15), par exemple

maximiser $z = 24 + \frac{7}{3} x_3 - \frac{1}{3} x_4$
sous les contraintes

$$
\begin{aligned}
x_1 &= 2 + \tfrac{1}{3} x_3 - \tfrac{1}{3} x_4 \\
x_2 &= 4 - \tfrac{5}{3} x_3 + \tfrac{1}{3} x_4 \\
x_5 &= 0 - \tfrac{11}{3} x_3 + \tfrac{1}{3} x_4 \\
x_1 &\geqslant 0 ; \; x_2 \geqslant 0 ; \; x_3 \geqslant 0 \\
x_4 &\geqslant 0 ; \; x_5 \geqslant 0
\end{aligned}
\qquad (15)
$$

x_3 entre dans la base et x_5 en sort. La valeur de x_3 est limitée à 0 par l'expression de x_5. Après **pivotement**, on obtient

maximiser $z = 24 + \frac{1}{11} x_5 - \frac{7}{11} x_4$
sous les contraintes

$$
\begin{aligned}
x_1 &= 0 + \tfrac{1}{11} x_5 - \tfrac{4}{11} x_4 \\
x_2 &= 2 - \tfrac{5}{11} x_5 - \tfrac{1}{11} x_4 \\
x_3 &= 4 + \tfrac{3}{11} x_5 + \tfrac{3}{11} x_4 \\
x_1 &\geqslant 0 ; \; x_2 \geqslant 0 ; \; x_3 \geqslant 0 \\
x_4 &\geqslant 0 ; \; x_5 \geqslant 0
\end{aligned}
\qquad (16)
$$

La valeur des variables n'a pas changé; celle de la fonction objective non plus. Ceci est fâcheux parce qu'on aurait voulu voir croître la fonction objective à chaque pivotement.

Ici, cela ne prête pas à conséquence parce que si l'on pivote à nouveau, on obtient une nouvelle base non dégénérée et la valeur de z croît strictement.

Exercice
Effectuer les calculs.

Quand la solution de base présente est dégénérée, il peut arriver que le prochain pivotement fasse apparaître à nouveau une solution de base dégénérée. Et cela peut se répéter un grand nombre de fois. Si deux bases de cette suite sont identiques, on est en présence d'un cycle de solutions de base dégénérées. En présence de cyclage, on peut pivoter indéfiniment, sans améliorer la valeur de z.

Le cyclage apparaît rarement en pratique et peut être contré par la façon de choisir les variables entrantes et sortantes. La règle de Bland offre un exemple d'une telle méthode.

Exercice (cf. Chvátal [10])
Soit le programme linéaire

$$
\begin{aligned}
&\text{maximiser} \quad z = \quad 10x_1 - 57x_2 - 9x_3 - 24x_4 \\
&\text{sous les contraintes} \\
&x_5 = \quad -0{,}5x_1 + 5{,}5x_2 + 2{,}5x_3 - 9x_4 \\
&x_6 = \quad -0{,}5x_1 + 1{,}5x_2 + 0{,}5x_3 - x_4 \qquad (17) \\
&x_7 = 1 - \quad x_1 \\
&x_i \geqslant 0, \; i = 1, \ldots, 7
\end{aligned}
$$

Appliquer la méthode du simplexe à ce programme en faisant usage des règles suivantes:
— la variable entrante est la variable hors-base dont le coefficient dans z est le plus élevé.
— si plusieurs variables de base sont candidates pour sortir de la base, choisir celle dont l'indice est le plus petit.

(i) Montrer qu'après la 6ème itération, on obtient la même base que la base de départ.
(ii) Prédire quel sera l'état du système après la 7ème itération, puis après la 22ème itération.
(iii) Appliquer la méthode du simplexe couplée à la règle de Bland à (17). Sans effectuer aucun calcul, on peut dire que les 5 premières itérations vont donner les mêmes résultats que précédemment; expliquer pourquoi. Quel est maintenant le résultat de la 6ème, puis de la 7ème itération?

(v) Construction d'une solution réalisable initiale
Soit le programme linéaire

$$
\begin{aligned}
&\text{maximiser} \quad c_1x_1 + c_2x_2 + \ldots + c_nx_n \\
&\text{sous les contraintes} \\
&a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \leq b_1 \\
&a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \leq b_2 \qquad (18) \\
&\text{----------} \\
&a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n \leq b_m \\
&x_1 \geqslant 0; \; x_2 \geqslant 0; \ldots; \; x_n \geqslant 0
\end{aligned}
$$

Si les b_i sont tous **non-négatifs**, alors le n-uplet (0, 0, ... , 0) est une solution réalisable.

Quand certains des b_i sont négatifs, il y a deux questions qui se posent:
— le programme possède-t-il une solution réalisable?
— si oui, en trouver une.

Il serait donc intéressant de trouver une procédure qui
— ou bien nous indique que le programme ne possède aucune solution réalisable
— ou bien exhibe une telle solution.

Considérons la première inégalité de (18), soit

$$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \leq b_1 \tag{19}$$

et donnons des valeurs arbitraires $\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n$ à $x_1, x_2, \ldots, x_n$. Pour savoir si (19) est satisfaite pour ces valeurs des variables x_i, il suffit, bien sûr, d'effectuer les calculs et de comparer les deux membres. Nous allons nous y prendre autrement.

Si (19) est satisfaite pour ces valeurs des variables, il en est de même a fortiori de l'inégalité

$$a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n + t \leq b_1 \tag{20}$$

où t est une variable à laquelle on a attribué une valeur non positive t quelconque. De plus même si (19) n'est pas satisfaite, (20) le sera encore si on attribue à t une valeur négative suffisamment grande en valeur absolue, c'est-à-dire si

$$t \leq b_1 - (a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n)$$

Dans le 1er cas, n'importe quelle valeur non-positive de t, y compris zéro, fera l'affaire. Dans le 2ème cas, t devra être strictement négative. On pourrait donc procéder de la façon suivante. Donner à t une valeur très grande en valeur absolue ($t = -\infty$, idéalement) et augmenter la valeur de t tout en s'assurant que (20) demeure satisfaite. Si on peut se rendre jusqu'à la valeur 0, c'est que (19) est satisfaite pour les valeurs $\bar{x}_1, \bar{x}_2, \ldots, \bar{x}_n$. Sinon, (19) ne l'est pas.

Nous venons en fait d'exprimer un problème de maximisation sous contrainte:

$$\begin{aligned} &\text{maximiser } t \\ &\text{sous les contraintes} \\ &a_{11}\bar{x}_1 + a_{12}\bar{x}_2 + \ldots + a_{1n}\bar{x}_n + t \leq b_1 \\ &t \leq 0 \end{aligned} \tag{21}$$

Ce raisonnement vaut pour l'ensemble des inégalités de (18). De plus, on n'est pas obligé de fixer les variables $x_1, x_2, \ldots, x_n$, ce qui est heureux parce que, justement, à ce stade-ci, on ne sait pas à quelle valeur il faut les fixer.

Finalement, la contrainte de non-positivité de la variable t peut se transformer en contrainte de non-négativité, grâce au changement de variable $t = -x_0$.

Le problème auxiliaire

$$\begin{aligned} &\text{maximiser } w = -x_0 \\ &\text{sous les contraintes} \\ &\begin{array}{l} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n - x_0 \leq b_1 \\ \hline a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n - x_0 \leq b_m \end{array} \\ &x_1 \geq 0;\ x_2 \geq 0;\ \ldots;\ x_n \geq 0;\ x_0 \geq 0 \end{aligned} \tag{22}$$

permet de répondre aux deux questions posées au début de cette section.

Si $(\bar{x}_0 = 0 , \bar{x}_1 , \bar{x}_2 , \ldots , \bar{x}_n)$ est une solution optimale de (22), alors $(\bar{x}_1 , \bar{x}_2 , \ldots , \bar{x}_n)$ est une solution réalisable de (18). Si la valeur optimale de w est négative, (18) ne possède aucune solution réalisable.

On est donc ramené à résoudre d'abord (22). Mais tous ces développements n'auront servi à rien si (22) ne possède pas une solution réalisable facilement identifiable. Heureusement, il suffit de choisir $x_1 = x_2 = \ldots = x_n = 0$ et $x_0 = - \min \{ b_1 , b_2 , \ldots , b_m \}$

Appliquons ce raisonnement à un exemple numérique. Soit

$$
\begin{aligned}
&\text{maximiser } z = -2x_1 + 3x_2 - x_3 \\
&\text{sous les contraintes} \\
(23) \qquad &\begin{aligned} -2x_1 + x_2 + x_3 &\leq -4 \\ x_1 + x_2 - 2x_3 &\leq 1 \\ -3x_1 + 4x_2 + x_3 &\leq -6 \end{aligned} \\
&x_1 \geq 0 ; x_2 \geq 0 ; x_3 \geq 0
\end{aligned}
$$

(0, 0, 0) n'est pas une solution réalisable de (23). Soit le programme auxiliaire

$$
\begin{aligned}
&\text{maximiser } w = -x_0 \\
&\text{sous les contraintes} \\
(24) \qquad &\begin{aligned} -2x_1 + x_2 + x_3 - x_0 &\leq -4 \\ x_1 + x_2 - 2x_3 - x_0 &\leq 1 \\ -3x_1 + 4x_2 + x_3 - x_0 &\leq -6 \end{aligned} \\
&x_1 \geq 0 ; x_2 \geq 0 ; x_3 \geq 0 ; x_0 \geq 0
\end{aligned}
$$

En introduisant les variables d'écart, on obtient

$$
\begin{aligned}
&\text{maximiser } w = -x_0 \\
&\text{sous les contraintes} \\
(25) \qquad &\begin{aligned} x_4 &= -4 + 2x_1 - x_2 - x_3 + x_0 \\ x_5 &= 1 - x_1 - x_2 + 2x_3 + x_0 \\ x_6 &= -6 + 3x_1 - 4x_2 - x_3 + x_0 \end{aligned} \\
&x_0 \geq 0 ; x_i \geq 0 , i = 1 , \ldots , 6
\end{aligned}
$$

(25) définit un dictionnaire non réalisable. En effet, la solution correspondante, soit $(x_0 = 0,0,0,0 , x_4 = -4 , x_5 = 1 , x_6 = -6)$ n'est pas réalisable. Mais on a vu qu'une solution réalisable de (24) est donnée par $x_1 = x_2 = x_3 = 0$ et $x_0 = 6$. En faisant entrer x_0 dans la base, on transforme (25) en un dictionnaire réalisable:

$$
\begin{aligned}
&\text{maximiser } w = -6 + 3x_1 - 4x_2 - x_3 - x_6 \\
&\text{sous les contraintes} \\
(26) \qquad &\begin{aligned} x_0 &= 6 - 3x_1 + 4x_2 + x_3 + x_6 \\ x_4 &= 2 - x_1 + 3x_2 + x_6 \\ x_5 &= 7 - 4x_1 + 3x_2 + 3x_3 + x_6 \end{aligned} \\
&x_0 \geq 0 ; x_i \geq 0 , i = 1 , \ldots , 6
\end{aligned}
$$

On applique l'algorithme du simplexe à (26) jusqu'à l'optimalité. On adopte toutefois la règle suivante: si x_0 est en compétition avec d'autres variables pour sortir de la base, c'est x_0 qui est choisie comme variable sortante. Deux cas sont alors possibles:

— ou bien x_0 est une variable de base dans la solution de base optimale trouvée. Alors, $x_0 > 0$ et le problème original ne possède pas de solution réalisable.

— ou bien x_0 est une variable hors-base dans la solution de base optimale trouvée. Alors, $x_0 = 0$

Exercice
Résoudre (26).

3.4.7 Résumé

Algorithme du simplexe

Données
Un programme linéaire écrit sous la forme

$$\begin{aligned} &\max\ z = c_1x_1 + c_2x_2 + \ldots + c_nx_n \\ &\text{sous les contraintes} \\ &a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n \leq b_1 \\ &a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n \leq b_2 \\ &\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad (27) \\ &a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n \leq b_m \\ &x_1 \geqslant 0\,;\ x_2 \geqslant 0\,;\ \ldots\,;\ x_n \geqslant 0 \end{aligned}$$

Résultats
Ou bien
— on indique que le problème ne possède pas de solution réalisable.
— on indique qu'il ne possède pas de solution bornée
— on fournit une solution optimale au problème.

Phase I

Données
Le programme linéaire (27)

Résultats
— ou bien on indique que (27) ne possède pas de solution réalisable
— ou bien on fournit une solution réalisable $(x_1', x_2', \ldots, x_n')$ de (27).

Si on connaît une solution réalisable de (27) — par exemple, si (0, 0, ..., 0) est une solution réalisable de (27) — aller à la phase II.

Sinon
— définir un programme linéaire auxiliaire:

$$\begin{aligned} &\max\ w = -x_0 \\ &\text{sous les contraintes} \\ &a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n - x_0 \leq b_1 \\ &a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n - x_0 \leq b_2 \\ &\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad\qquad (28) \\ &a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n - x_0 \leq b_m \\ &x_1 \geqslant 0\,;\ x_2 \geqslant 0\,;\ \ldots\,;\ x_n \geqslant 0\,;\ x_0 \geqslant 0 \end{aligned}$$

— (28) possède une solution réalisable:

$$\begin{aligned} &x_0 = -\min\{b_1, b_2, \ldots, b_m\} \\ &x_1 = x_2 = \ldots = x_n = 0 \end{aligned} \qquad (29)$$

— Appliquer la phase II à (28) et (29).

On obtient une solution optimale $(\overline{x_0}, \overline{x_1}, \overline{x_2}, \ldots, \overline{x_n})$.
- Si $\overline{x_0} < 0$, on conclut que (27) ne possède pas de solution réalisable et on s'arrête.
- Si $\overline{x_0} = 0$, alors $(\overline{x_1}, \overline{x_2}, \ldots, \overline{x_n})$ est une solution réalisable de (27). On applique la phase II à (27) et à $(\overline{x_1}, \overline{x_2}, \ldots, \overline{x_n})$.

Phase II

Données

- Un programme linéaire

$$\begin{aligned} &\max\ z = c_1x_1 + \ldots + c_nx_n \\ &\text{sous les contraintes} \\ &\left.\begin{array}{l} a_{11}x_1 + \ldots + a_{1n}x_n \leqslant b_1 \\ a_{21}x_1 + \ldots + a_{2n}x_n \leqslant b_2 \\ \hline a_{m1}x_1 + \ldots + a_{mn}x_n \leqslant b_m \end{array}\right\} \quad (30) \\ &x_1 \geqslant 0;\ \ldots;\ x_n \geqslant 0 \end{aligned}$$

- Une solution réalisable $(\overline{x_1}, \overline{x_2}, \ldots, \overline{x_n})$ de (30).

Résultat

— Ou bien on indique que (30) ne possède pas de solution optimale finie
— ou bien on fournit une solution optimale.

Étape 1

Introduire m variables d'écart $x_{n+1}, x_{n+2}, \ldots, x_{n+m}$ de façon à écrire les contraintes ordinaires du problème sous la forme d'équations. (30) est équivalent à

$$\begin{aligned} &\text{maximiser}\ z = c_1x_1 + c_2x_2 + \ldots + c_nx_n \\ &\text{sous les contraintes} \\ &\left.\begin{array}{l} a_{11}x_1 + a_{12}x_2 + \ldots + a_{1n}x_n + x_{n+1} = b_1 \\ a_{21}x_1 + a_{22}x_2 + \ldots + a_{2n}x_n + x_{n+2} = b_2 \\ \hline a_{m1}x_1 + a_{m2}x_2 + \ldots + a_{mn}x_n + x_{n+m} = b_m \end{array}\right\} \quad (31) \\ &x_1 \geqslant 0;\ x_2 \geqslant 0;\ \ldots;\ x_{n+m} \geqslant 0 \end{aligned}$$

Soit $x^0 = (x_1 = \overline{x_1}, \ldots, x_n = \overline{x_n}, x_{n+1} = 0, \ldots, x_{n+m} = 0)$, une solution réalisable de (31). Écrire (31) sous la forme d'un dictionnaire réalisable (i.e. écrire les variables de base et la fonction objective en fonction des variables hors-base).

Étape 2

Si dans l'expression de z tous les coefficients des variables hors-base sont non positifs, la solution présente est optimale.
Sinon, **choisir** une variable dont le coefficient est positif. Soit x_s la variable choisie. Si le coefficient de x_s est non négatif dans **chacune** des équations du présent dictionnaire, on conclut que le programme linéaire donné n'est pas borné et on s'arrête.
Sinon, aller à l'étape 3.

Étape 3 (Pivotement)

Chaque contrainte d'égalité impose une limite à l'accroissement de x_s. Soit a, le terme constant d'une certaine contrainte d'égalité; a est non négatif. Soit b, le coefficient de x_s dans cette contrainte. Si b est négatif, l'accroissement de x_s est limité à la valeur $\frac{a}{-b}$.

Identifier les équations pour lesquelles la valeur $\frac{d}{-b}$ est la plus petite possible. **Choisir** une de ces équations. Elle correspond à une variable de base x_r . x_r sort de la base et y est remplacée par x_s : on écrit x_s en fonction de x_r et des autres variables hors-base. On remplace x_s par son expression dans toutes les autres contraintes d'égalité et dans la fonction objective. Retour à l'étape 2.

Fin

Remarque
Le choix de la variable entrante et de la variable sortante répond à des considérations d'efficacité ou encore au souci d'éviter un éventuel cyclage.

3.4.8 Retour à la modélisation de problèmes

Présenter quelques exemples simples dont la résolution se ramène à minimiser une fonction sous contraintes linéaires. Varier les exemples de façon à faire apparaître des contraintes de la forme $a_{i1}x_1 + \ldots + a_{in}x_n \geqslant b_i$.
et aussi $a_{i1}x_1 + \ldots + a_{in}x_n = b_i$.

$x_k \leqslant 0$.

Maintenant, on est en mesure de montrer aux étudiants que tout problème **d'optimisation d'une fonction linéaire sous contraintes linéaires** peut se ramener à la forme standard définie à la section 3.4.3.

a) **minimiser** $c_1x_1 + c_2x_2 + \ldots + c_nx_n$ **est équivalent** à maximiser
$(-c_1)x_1 + (-c_2)x_2 + \ldots + (-c_n)x_n$

b) La contrainte $a_{i1}x_1 + \ldots + a_{in}x_n \geqslant b_i$
est équivalente à $(-a_{i1})x_1 + (-a_{i2})x_2 + \ldots + (-a_{in})x_n \leqslant -b_i$.

c) La contrainte $a_{i1}x_1 + \ldots + a_{in}x_n = b_i$ est équivalente à l'ensemble des deux contraintes

$$\begin{cases} a_{i1}x_1 + \ldots + a_{in}x_n \geqslant b_i \\ a_{i1}x_1 + \ldots + a_{in}x_n \leqslant b_i \end{cases}$$

d) La contrainte $x_k \leqslant 0$ est équivalente à

$$\begin{cases} t = -x_k \\ t \geqslant 0 \end{cases}$$

Il suffit de remplacer x_k par $-t$ partout dans le programme et de contraindre t à être non-négative.

e) Il peut arriver qu'une variable x ne soit soumise à aucune contrainte de signe. x peut alors s'écrire $x = \hat{x} - \bar{x}$, où $\hat{x}$ et $\bar{x}$ sont contraintes à être non-négatives.

f) Les contraintes $x \leqslant u$ ($u \neq 0$) et $x \geqslant l$ ($l \neq 0$) peuvent être assimilées à des contraintes ordinaires.
La paire de contraintes $l \leqslant x \leqslant u$ (avec $l < u$) est **équivalente** à $0 \leqslant x - l \leqslant u - l$ (avec $u - l > 0$). Dans ce cas on remplace x par $y + l$ et la nouvelle variable est soumise à la contrainte "ordinaire" $y \leqslant u - l$ et à la contrainte de signe $y \geqslant 0$.

3.4.9 Complexité de la méthode du simplexe

(voir section 2.4 et aussi [10])
La méthode du simplexe offre un exemple remarquable d'un algorithme que les calculs d'efficacité théorique donnent pour mauvais alors qu'il est excellent en pratique.
En effet, il est connu que pour des problèmes **pratiques** de la forme

$$\begin{aligned} &\text{maximiser } c_1x_1 + \ldots + c_nx_n \\ &\text{sous les contraintes} \\ &a_{11}x_1 + \ldots + a_{1n}x_n \leq b_1 \\ &\qquad\qquad \vdots \\ &a_{m1}x_1 + \ldots + a_{mn}x_n \leq b_m \\ &x_1 \geqslant 0 , \ldots , x_n \geqslant 0 \end{aligned}$$

le nombre moyen d'itérations tourne autour de $\frac{3m}{2}$.

En ce qui concerne la complexité théorique de cet algorithme selon le critère du pire-des-cas, il existe des programmes linéaires de n inéquations à n variables pour lesquels la méthode du simplexe passe à travers $2^n - 1$ itérations.

3.5 Calculer le déterminant d'une matrice carrée d'ordre n.

3.5.1 Utilisation

Math 105, Math 205

3.5.2 1ère Méthode: Appliquer la définition

Par définition, $\det(A) = \sum_{\sigma} \operatorname{sgn}(\sigma)\, a_{1\sigma_1} \ldots a_{n\sigma_n}$ où σ représente une permutation de $\{1, 2, \ldots, n\}$ et $\operatorname{sgn}(\sigma)$ représente la signature de cette permutation.

Complexité: Il faut faire la somme de n! nombres, chacun d'eux étant un produit de n facteurs. Donc il faut effectuer au plus $n!\,(n-1)$ multiplications et $n! - 1$ additions.

3.5.3 2ème Méthode: Mettre la matrice sous forme échelonnée

Soit à calculer det (A). On applique à la matrice A la méthode de réduction de Gauss (voir Algorithme 1, Section 3.1) de façon à la mettre sous forme échelonnée. Soit E, le résultat de cette opération. Alors

$$\det(E) = (-1)^t \cdot k_1 \cdot k_2 \cdot \ldots \cdot k_r \cdot \det(A)$$

où t est le nombre de fois où on a permuté deux lignes de A au cours de sa réduction et $k_1 , k_2 , \ldots , k_r$ sont les multiplications effectuées sur les lignes de A au cours de ce processus.

Donc, $\det(A) = \frac{(-1)^t}{k_1 \cdot k_2 \cdot \ldots \cdot k_r} \cdot \det(E)$, et det (E) est égal au produit des éléments de la diagonale principale de E.

Complexité: Au plus $\frac{n(n+1)(2n+1)}{6}$ additions et multiplications, plus $\frac{n(n+1)}{2}$ divisions.

3.5.4 Méthode ad hoc

Utiliser — quand c'est possible — la structure de A et les propriétés de la fonction déterminant de façon à obtenir rapidement det (A).

3.6 Développement du déterminant d'une matrice carrée d'ordre 4

Référence: [39]

3.6.1 Utilisation

Math 105, Math 205

3.6.2 Le problème

On sait que le calcul d'un déterminant d'ordre 2 ou d'ordre 3 peut s'effectuer selon la règle simple suivante:

ou $a_{11} \cdot a_{22} - a_{12} \cdot a_{21}$

ou

$$\begin{aligned} &a_{11}\ a_{22}\ a_{33} + a_{12}\ a_{23}\ a_{31} + a_{13}\ a_{21}\ a_{32} \\ &+ a_{13}\ a_{22}\ a_{31} - a_{11}\ a_{23}\ a_{32} - a_{12}\ a_{21}\ a_{33} \end{aligned}$$

Une question qui revient souvent, c'est celle-ci:

Est-il possible d'évaluer un déterminant d'ordre 4 par une méthode analogue?

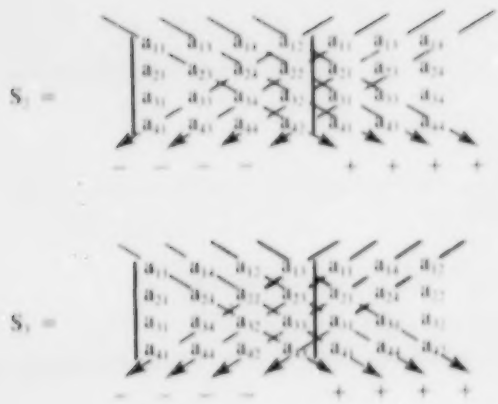
3.6.3 Une méthode

Un premier élément de réponse: Il ne suffit pas d'adjoindre les trois premières colonnes à la suite des quatre colonnes de la matrice et de procéder comme pour un déterminant d'ordre 3. En effet, on n'obtiendrait que 8 termes de cette façon alors qu'il y en a 4!, c'est-à-dire 24.

Soit $A = \begin{bmatrix} a_{11} & a_{12} & a_{13} & a_{14} \\ a_{21} & a_{22} & a_{23} & a_{24} \\ a_{31} & a_{32} & a_{33} & a_{34} \\ a_{41} & a_{42} & a_{43} & a_{44} \end{bmatrix}$. Dans [39] on montre que $\det(A) = S_1 + S_2 + S_3$

avec

$S_1 =$



Commentaire
Ceci constitue un bon exemple de recherche mathématique accessible à un étudiant de CEGEP et pourrait faire l'objet d'un travail dirigé.

3.7 Détermination d'un ensemble de vecteurs linéairement indépendants, de poids maximum

Référence: [7]

3.7.1 Utilisation

Math 205

3.7.2 Définition du problème

Soit E un espace vectoriel. Soit S une famille finie de vecteurs de E. On définit sur S une fonction qui à chaque vecteur v de S associe un "poids", $p(v) > 0$.
Trouver une sous-famille S' de S, linéairement indépendante, dont le poids est maximum.

3.7.3 Un algorithme de résolution

Algorithme

Étape 1
Classer les vecteurs de S par ordre de poids décroissant.

Étape 2
Choisir les vecteurs un à la fois, dans l'ordre de poids décroissant. Un vecteur est rejeté quand il ne forme pas un système linéairement indépendant avec les vecteurs déjà choisis. On s'arrête quand tous les vecteurs ont été examinés.

Fin

Remarque
Le problème de la recherche d'un sous-ensemble de S, linéairement indépendant et de cardinalité maximale (voir section 3.3) constitue un cas particulier du présent problème. Il correspond à la condition $p(v) = 1$, pour tout vecteur v de S.

Justification de l'algorithme Voir section 3.13.

3.8 Arithmétique Élémentaire et informatique

Utilisation

Info 101, Info 201, Info 248, **Cours d'introduction à l'informatique**

Introduction

L'enseignement des algorithmes d'addition, de soustraction, de multiplication et de division aux jeunes enfants répond d'abord à la nécessité de leur fournir rapidement un outil de calcul efficace. L'accent y est mis — par nécessité — sur l'application d'une règle plutôt que sur la compréhension des mécanismes de la numération dans une base donnée. Il est sans doute normal de procéder ainsi vu le jeune âge des élèves mais il faudrait revenir sur ces méthodes quelques années plus tard pour les analyser et en comprendre le fonctionnement.
Un cours d'introduction à l'informatique offre l'occasion d'une telle révision grâce à l'étude de la numération en base 2. À cette occasion, on peut réfléchir sur la numération décimale, explorer les liens entre les différents modes de représentation numérique et justifier les algorithmes usuels de l'arithmétique élémentaire.

Numération en base b
Soit b, un entier supérieur ou égal à 2. Tout nombre entier z non négatif peut s'écrire de façon unique sous la forme

$$z = a_n b^n + a_{n-1} b^{n-1} + \ldots + a_2 b^2 + a_1 b + a_0 ,$$

où les coefficients a_i sont des nombres entiers qui vérifient $0 \leq a_i \leq b - 1$. On écrit $z = \overline{a_n a_{n-1} \ldots a_1 a_0}^b$. Quand b = 10, on écrit $z = a_n a_{n-1} \ldots a_1 a_0$.

3.8.1 Passage de la numération décimale à la numération en base b (b ≠ 10)

L'expression de z en base b est complètement déterminée par la connaissance de

$$a_n , a_{n-1} , \ldots , a_1 , a_0$$

Pour déterminer ces coefficients, on procède à des divisions successives par b. En effet, soit $z = a_n b^n + a_{n-1} b^{n-1} + \ldots + a_1 b + a_0$. (Pour le moment, cette écriture est purement formelle). Divisons z par b, $z = bq_0 + r_0$, avec $0 \leq r_0 \leq b - 1$. q_0 et r_0 sont respectivement le quotient et le reste de la division de z par b.

$$q_0 = a_n b^{n-1} + a_{n-1} b^{n-2} + \ldots + a_2 b + a_1$$
$$r_0 = a_0$$

Pour déterminer a_1 on divise q_0 par b:

$$q_1 = a_n b^{n-2} + a_{n-1} b^{n-3} + \ldots + a_3 b + a_2$$
$$r_1 = a_1$$

On procède ainsi de proche en proche jusqu'à

$q_{n-2} = a_n b + a_{n-1}$ $\qquad$ $q_{n-1} = a_n$

$r_{n-2} = a_{n-2}$ $\qquad$ $r_{n-1} = a_{n-1}$

Algorithme: Passage de la base 10 à la base b

Étape 1
Mettre le compteur I à zéro

Étape 2
Supposons que le compteur I contienne la valeur i. Diviser z par b. Soit q le quotient et r le reste de la division.
Poser $a_i = r$ et $z = q$.Si $z = 0$, on s'arrête. Sinon, augmenter la valeur de I et aller au début de l'étape 2.

Fin

Complexité
Environ t divisions et additions, où $t = \lceil \log_b z \rceil$, l'approximation entière de $\log_b z$ par excès.

3.8.2 Passage de la numération en base b (b ≠ 10) à la numération décimale.

$z = a_n b^n + a_{n-1} b^{n-1} + \ldots + a_1 b + a_0$. Les coefficients $a_0, a_1, \ldots, a_n$ sont connus. Il reste à calculer $b^2, b^3, \ldots, b^n$ et à effectuer le calcul de z.

Complexité
$2n - 1$ multiplications, plus n additions. Ces calculs peuvent être simplifiés grâce à la "règle de Horner".

3.8.3 La règle de Horner

C'est une méthode rapide de calcul de $f(x) = a_n x^n + a_{n-1} x^{n-1} + \ldots + a_1 x + a_0$. On a

$$f(x) = (\ldots((((a_n)x + a_{n-1})x + a_{n-2})x + a_{n-3})x + \ldots + a_1)x + a_0$$

Exercice
Exprimer la règle de Horner sous la forme d'un algorithme.

Complexité
n additions et multiplications

3.8.4 Soustraction par la méthode du complément

Le Problème
Soit à calculer x − y, avec $x = \overline{1\,1\,0\,1\,0\,0\,1}^{\,2}$ et $y = \overline{1\,0\,1\,1\,1}^{\,2}$.
On peut procéder comme dans le système décimal:

"emprunts"		0	11	0	1 1	11	
1	1	0	1	0	0	1	
		1	0	1	1	1	
1	0	1	0	0	1	0	

On peut aussi procéder autrement, comme on va le voir.

Méthode du complément

i) Écrire $\overline{1\,0\,1\,1\,1}^{\,2}$ sous la forme $\overline{0\,0\,1\,0\,1\,1\,1}^{\,2}$, de façon à ce que les deux nombres aient autant de chiffres l'un que l'autre.

ii) Complémenter $\overline{0\,0\,1\,0\,1\,1\,1}^{\,2}$ en remplaçant 0 par 1 et 1 par 0. On obtient $z = \overline{1\,1\,0\,1\,0\,0\,0}^{\,2}$.

iii) Additionner x et z.

$$\begin{array}{r} 1\,1\,0\,1\,0\,0\,1 \\ +\;1\,1\,0\,1\,0\,0\,0 \\ \hline 1\,1\,0\,1\,0\,0\,0\,1 \end{array}$$

iv) Additionner 1 au résultat et supprimer le chiffre le plus à gauche:

$$\begin{array}{r} 1\,1\,0\,1\,0\,0\,0\,1 \\ +\qquad\qquad\;\;1 \\ \hline 1\,0\,1\,0\,0\,1\,0 \end{array}$$

On a: $x + y = \overline{1\,0\,1\,0\,0\,1\,0}^{\,2}$.

La méthode du complément permet d'éviter les "emprunts" et ramène un problème nouveau (soustraction) à un problème que l'on sait déjà résoudre (addition).
Elle est indépendante du système de numération choisi. Ainsi, dans le système décimal, le calcul 24580 − 7395 s'effectue ainsi:

i) 7395 est écrit 07395

ii) 07395 est changé en 92604
(Le complément de i est 9 − i, pour i = 0, 1, 2, ... , 9)

iii) 24580 + 92604 = 117184

iv) Le résultat cherché: 17185

Justification de la méthode

On va montrer comment cela marche sur un exemple numérique. La généralisation est immédiate.

On a: $24580 - 7395 = 24580 - 07395$

$= 24580 - 07395 + 92604 - 92604$

$= (24580 + 92604) - (07395 + 92604)$

$= 117184 - 99999.$

Cette dernière opération serait très simple si le deuxième nombre était 100 000 plutôt que 99999.

On a: $117184 - 99999 = 117184 - 99999 + 1 - 1$

$= (117184 + 1) - (99999 + 1) = 117185 - 100\,000$

$= 17185$

Exercice

Montrer que le chiffre que l'on supprime en iv) est toujours "1", quels que soient les nombres x et y et quelle que soit la base.

3.8.5 Algorithme d'Euclide

Soit a et b, deux nombres entiers non négatifs. On désigne leur plus grand diviseur commun par $a \wedge b$.
On a: $a \wedge 1 = 1$; $a \wedge a = a$; $a \wedge b = b \wedge a$
$a \wedge a = a$
Par convention, $0 \wedge 0 = 0$

Méthode de calcul
Soit à calculer $3258 \wedge 120$
On divise 3258 par 120 et on trouve un reste de 18.
On divise 120 par 18. Il reste 12
On divise 18 par 12. Il reste 6
On divise 12 par 6. Il reste 0
$6 = 3258 \wedge 120$

	27	6	1	2
3258	120	18	12	6
018	12	6	0	

Justification
Soit à calculer $a \wedge b$ avec $a \geq b > 0$. On divise a par b. On a: $a = bq_0 + r_0$, avec $0 \leq r_0 \leq b - 1$; q_0 et r_0 sont respectivement le quotient et le reste de la division.

Si x est un diviseur commun de a et b, alors x divise aussi r_0, parce que $r_0 = a - bq_0$. Inversement, si x est diviseur commun de b et r_0, alors x divise a. Donc, les couples (a, b) et (b, r_0) ont le même ensemble de diviseurs. Par suite, $a \wedge b = b \wedge r_0$.

Donc le calcul de $a \wedge b$ se ramène à celui de $b \wedge r_0$, où $0 \leq r_0 < b \leq a$. On procède ainsi, de proche en proche, jusqu'à ce que le reste de la division soit 0.

Alors $a \wedge b = b \wedge r_0 = r_0 \wedge r_1 = r_1 \wedge r_2 = \ldots = r_k \wedge r_{k+1}$, où $r_{k+1} = 0$. Donc $a \wedge b = r_k$.

Exercice
Écrire l'algorithme d'Euclide.

Une variante [20]
Divisons a par b. On a: $a = bq + r$, avec $0 \leq r \leq b - 1$ et aussi,

$$a = bq + r + b - b = b(q + 1) - (b - r), \text{ avec } 0 \leq b - r \leq b - 1.$$

$a \wedge b = b \wedge (b - r)$ pour les mêmes raisons qui font que $a \wedge b = b \wedge r$. Quand $b - r$ est inférieur à r, on accélère la convergence de la méthode en calculant $b \wedge (b - r)$ plutôt que $b \wedge r$. Ainsi, dans le cas de 3258 et 120, on a

	27	6	3
3258	120	18	[6]
(18)	(12)	(0)	

$r_0 = 18 ; r_1 = 12 ; r_0 - r_1 = 6 < r_1$

3.8.6 Multiplication Égyptienne; Algorithme du Paysan Russe

Références: [8], [20].

La méthode de multiplication égyptienne (appelée aussi algorithme du paysan russe) permet de multiplier deux nombres entiers quelconques en sachant seulement
— additionner deux nombres
— multiplier par 2

Principe

Soit x et y, deux nombres entiers

- Si y est pair, alors $\frac{y}{2}$ est un nombre entier et

  (i) $xy = (2x) \cdot \frac{y}{2}$

- Si y est impair, alors $y - 1$ est pair et

  (ii) $xy = x(y - 1) + x$.

On part de x et y.
À chaque itération, on applique (i) ou (ii) selon le cas. On s'arrête quand $y = 0$.

Application à un exemple numérique

$x = 41$ et $y = 27$

	x	y	Restes
On applique (ii)	41	27	
On applique (i)	41	26	41
On applique (ii)	82	13	
On applique (i)	82	12	82
On applique (i)	164	6	
On applique (ii)	328	3	
On applique (i)	328	2	328
On applique (ii)	656	1	
	656	0	656

On obtient: $41 \times 27 = 656 \times 0 + 656 + 328 + 82 + 41 = 1107$

On peut simplifier la présentation des calculs:

Multiplication de x par 2	Division entière de y par 2
41	27
82	13
~~164~~	6
328	3
656	1
1107	

3.8.7 Calcul de X^Y, pour deux entiers X et Y

Soit à calculer X^Y, où X et Y sont deux entiers positifs. La solution évidente $X, X^2, X^3, \ldots, X^Y$ requiert $Y - 1$ multiplications. Il y a moyen de procéder beaucoup plus rapidement.

Une méthode semblable à la multiplication égyptienne [20]
Soit X, Y, Z, trois nombres entiers. Le produit $Z \cdot X^Y$ est invariant pour les deux opérations suivantes:

(i) quand Y est pair, on divise Y par 2 et on élève X au carré.
(ii) quand Y est impair, on le diminue d'une unité et on multiplie Z par X

Donc, $Z \cdot X^Y = Z \cdot (X^2)^{\frac{Y}{2}}$ quand Y est pair
$Z \cdot X^Y = ZX \cdot X^{Y-1}$ quand Y est impair.

Au départ, on fixe la valeur de Z à 1. On applique (i) et (ii) jusqu'à ce que $Y = 0$. Alors, Z donne le résultat cherché.

Exemple

$X^{21} = X \cdot X^{20} = X \cdot (X^2)^{10} = X\,(X^4)^5 = (X \cdot X^4)\,(X^4)^4$

$= X^5\,(X^8)^2 = X^5\,(X^{16})^1 = (X^5 \cdot X^{16})\,(X^{16})^0$

Ce calcul ne requiert que 5 multiplications au lieu de 20: calcul de

X^2, de X^4, de $X \cdot X^4 = X^5$, de X^{16} et de $X^5 \cdot X^{16} = X^{21}$

3.9 Calcul approché d'Aires; Intégration Numérique

3.9.1 Utilisation

Math 103; Math 203; Math 248; Math 408

3.9.2 Références

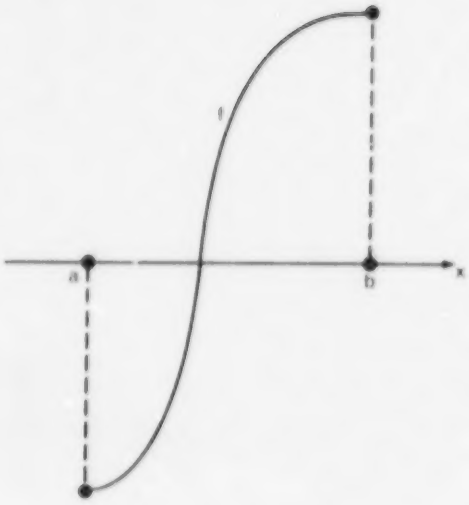
[9], [20], [51]

3.9.3 Définition du problème

Soit f, une fonction réelle définie sur $[a, b] \subset \mathbb{R}$.

Soit $I = \int_a^b f(x)\,dx$

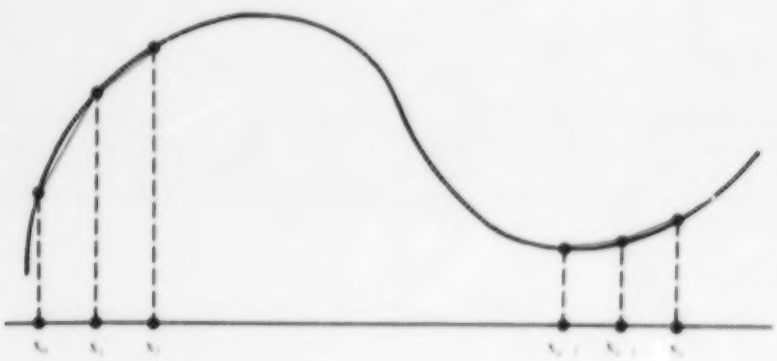
I est égale à l'aire de la surface comprise entre la courbe, l'axe des x et les droites parallèles d'équation x = a et x = b.



Il arrive que f ne soit pas intégrable et que l'on doive se contenter d'une valeur approchée de I. Trois méthodes sont couramment utilisées. Dans les trois cas, l'intervalle $[a, b]$ est divisé en n sous-intervalles d'égale longueur ($n \geq 1$). Supposons que les points de division soient

$$a = x_0 < x_1 < x_2 < \ldots < x_n = b$$

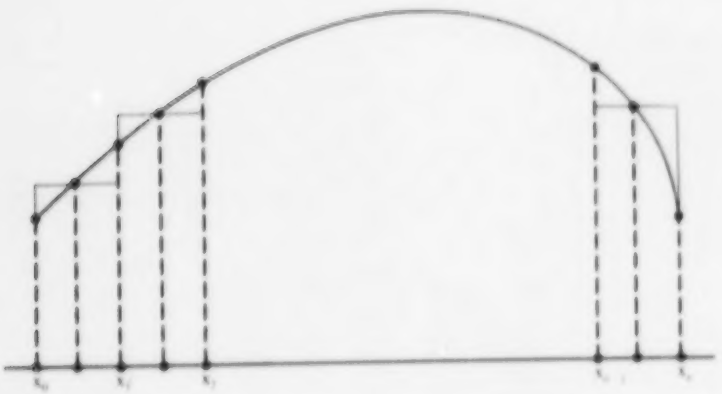
3.9.4 Méthode des Trapèzes



$$T_n(f) = \frac{(b - a)}{2n} \left[f(x_0) + 2f(x_1) + 2f(x_2) + \ldots + 2f(x_{n-1}) + f(x_n) \right]$$

3.9.5 Méthode de la tangente ou du point-milieu

$$M_n(f) = \frac{(b-a)}{n}\left[f\left(\frac{x_0 + x_1}{2}\right) + f\left(\frac{x_1 + x_2}{2}\right) + \ldots + f\left(\frac{x_{n-1} + x_n}{2}\right)\right]$$



3.9.6 Méthode de Simpson

Elle consiste à approximer la courbe par un arc de parabole dans chaque sous-intervalle. La valeur obtenue par cette méthode est une moyenne pondérée des deux valeurs qu'on vient de définir:

$$S_n(f) = \frac{T_n(f) + 2 \cdot M_n(f)}{3}$$

3.9.7 Analyse

Ces approximations sont d'autant meilleures que n est grand. En pratique, on part de n = 1 et, à chaque itération, on **double** le nombre de sous-intervalles. On calcule à chaque fois une majoration de l'erreur relative et on s'arrête quand cette majoration est "suffisamment petite". Quand on subdivise un intervalle en deux parties égales, les valeurs de T, M et S sont aisément mises à jour.

En effet, soit n = 1.

$$T_1(f) = (b-a)\left[\frac{f(a) + f(b)}{2}\right], \quad M_1(f) = (b-a) \cdot f\left(\frac{a+b}{2}\right)$$

Soit $x_0 = a < x_1 = \frac{a+b}{2} < x_2 = b$

$$T_1(f) = \frac{(b-a)}{2} \cdot \left[\frac{f(a) + f(x_1)}{2} + \frac{f(x_1) + f(b)}{2} \right]$$

$$= \frac{(b-a)}{2} \left[\frac{f(a) + f(b)}{2} \right] + \frac{(b-a)}{2} f(x_1)$$

$$= \frac{1}{2} [T_1(f) + M_1(f)]$$

$$M_2(f) = \frac{(b-a)}{2} \cdot f\left(\frac{x_0 + x_1}{2}\right) + \frac{(b-a)}{2} \cdot f\left(\frac{x_1 + x_2}{2}\right)$$

$$S_2(f) = \frac{T_2(f) + 2M_2(f)}{3}$$

En général, soit $x_0 = a < x_1 < x_2 < \ldots < x_{2n} = b$,

avec $n = 2^k$ (donc, $2 \cdot n = 2^{k+1}$).

$$T_{2n}(f) = \frac{1}{2} [T_n(f) + M_n(f)]$$

$$M_{2n}(f) = \frac{(b-a)}{2n} \left[f\left(\frac{x_0 + x_1}{2}\right) + \ldots + f\left(\frac{x_{2n-1} + x_{2n}}{2}\right) \right]$$

$$S_{2n}(f) = \frac{T_{2n}(f) + 2 \cdot M_{2n}(f)}{3}$$

L'erreur absolue de chacune des trois méthodes est définie par

$E(T_n) = | I - T_n(f) |$,

$E(M_n) = | I - M_n(f) |$ et

$E(S_n) = | I - S_n(f) |$

L'erreur relative est obtenue en divisant l'erreur absolue par $| I |$. **Les erreurs relatives ne sont pas connues mais elles sont généralement majorées par** $\frac{| T_n(f) - M_n(f) |}{| S_n(f) |}$.

On s'arrête quand ce nombre est inférieur à une valeur fixée d'avance, 10^{-6} **par exemple.**

3.9.8 Application: Dresser la table de la loi normale

Math 127; Math 190; Math 227; Math 257; Math 271; **Math 307**; Math 337; Math 90).

Calculer à l'aide d'un programme d'ordinateur $\int_0^h e^{-x^2} dx$ à 10^{-6} près, pour

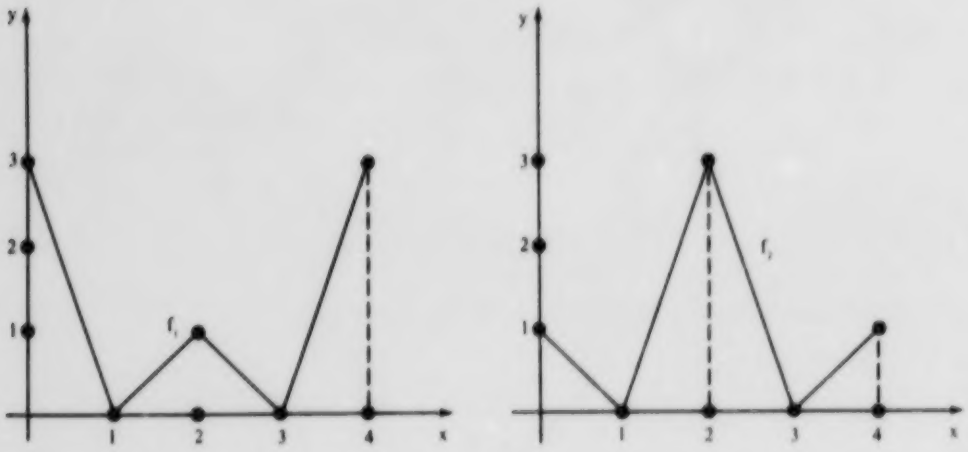
$$h = 0.01, 0.02, \ldots, 4.00$$

3.9.9 Comparaison des trois méthodes [51]

On sait qu'en général $E(S_n) < E(T_n) < E(M_n)$, c'est-à-dire, la règle de Simpson donne une meilleure approximation que la méthode des trapèzes et celle-ci, à son tour, est meilleure que la règle du point-milieu. Mais il n'en est pas toujours ainsi.

Par exemple, f_1 est telle que $I = 4$, $M_1(f_1) = 4$, $T_1(f_1) = 12$ et $S_1(f_1) = \frac{20}{3}$;

f_2 est telle que $I = 4$, $T_1(f_2) = 4$, $M_1(f_2) = 12$ et $S_1(f_2) = \frac{28}{3}$.



Mais il est impossible d'avoir à la fois $E(M_n) < E(S_n)$ et $E(T_n) \leq E(S_n)$

Théorème [51] Supposons que f, [a, b] et n soient tels que $E(M_n) < E(S_n)$. Alors on doit avoir $E(S_n) < E(T_n)$.

Preuve
Supposons que $E(M_n) < E(S_n)$ et $E(T_n) \leq E(S_n)$.

On a $S_n(f) = \frac{T_n(f) + 2 \cdot M_n(f)}{3}$

D'où, $E(S_n) = | 1 - S_n(f) | = | 1 - \frac{1}{3}T_n(f) - \frac{2}{3}M_n(f) |$

$= \frac{1}{3} | (1 - T_n(f)) + 2 (1 - M_n(f)) | \leq \frac{1}{3} | 1 - T_n(f) | + \frac{2}{3} | 1 - M_n(f) |$

$= \frac{E(T_n)}{3} + \frac{2\,E(M_n)}{3} < \frac{E(S_n)}{3} + \frac{2\,E(S_n)}{3}$

On a donc $E(S_n) < E(S_n)$, ce qui contredit l'hypothèse.

3.10 Méthode de recherche dichotomique

3.10.1 Introduction

Le jeu des 10 questions

Il s'agit d'un jeu très connu dont les règles se définissent comme suit:

Deux joueurs A et B sont en présence. A choisit un nombre x compris entre 1 et 1000, connu de lui seul. B doit déterminer la valeur de x. Pour cela on lui permet de poser jusqu'à 10 questions à A, de façon à pouvoir s'orienter à l'aide des réponses. Toutes les questions doivent être du type "Est-ce que x est inférieur ou égal à y?", où y est un nombre choisi par B. Le joueur A doit répondre par oui ou non à chaque question.

Voyons sur un exemple comment cela marche. Supposons que A choisisse le nombre 231. On aura alors la séquence suivante.

Questions	Réponses	Intervalle d'indétermination de x
		[1 , 1000]
$x \leq 500$?	Oui	[1 , 500]
$x \leq 250$?	Oui	[1 , 250]
$x \leq 125$?	Non	[126 , 250]
$x \leq 188$?	Non	[189 , 250]
$x \leq 219$?	Non	[220 , 250]
$x \leq 234$?	Oui	[220 , 234]
$x \leq 226$?	Non	[227 , 234]
$x \leq 230$?	Non	[231 , 234]
$x \leq 232$?	Oui	[231 , 232]
$x \leq 231$?	Oui	[231 , 231]

Analyse
À chaque étape, le nombre x se trouve localisé à l'intérieur d'un intervalle [a', b'] qui con-

tient b' − a' + 1 nombres. Le joueur B choisit y égal à la partie entière de $\frac{a' + b'}{2}$. B demande à A si x est inférieur ou égal à y. Si la réponse est oui, il sait que x appartient à l'intervalle [a', y]; si la réponse est non, x appartient à l'intervalle [y + 1 , b'] . Chaque réponse permet de réduire de moitié le nombre de valeurs numériques que contient l'intervalle d'indétermination.

Ainsi, après la n^{e} réponse, l'intervalle d'indétermination contient au plus $\left\lfloor \frac{b - a + 1}{2^n} \right\rfloor$ nombres, où [a, b] représente l'intervalle initial. Le nombre de questions nécessaires à la détermination de x est au plus égal à $n = \lceil \log_2(b - a + 1) \rceil$.

Commentaire

Cette méthode de recherche est très efficace à cause de sa rapidité de convergence.
Dans le cas discret (jeu des 10 questions, par exemple) elle permet d'obtenir la valeur cherchée au bout de n itérations ($n = \lceil \log_2 l \rceil$, où l est égal au nombre de valeurs numériques de l'intervalle d'indétermination initial). Dans le cas continu (voir les sections 3.10.3 et 3.10.4), cette méthode fournit, au bout de n itérations, un intervalle d'indétermination de largeur $\frac{l}{2^n}$, où l est la largeur de l'intervalle initial. Si l'on se fixe au préalable une marge d'incertitude égale à ε, on choisira une valeur de n telle que $\varepsilon \leqslant \frac{l}{2^n}$. (Il existe des variantes où l'intervalle d'indétermination n'est pas divisé en deux parties égales (voir la section 3.10.2)).

3.10.2 Calcul du logarithme décimal de x

Référence: [20]

Utilisation
Math 102; Math 122; Math 211; Math 408

Analyse du problème
Désignons par log x, le logarithme décimal de x.
On sait que:

- log 1 = 0
- log 10 = 1
- $\log \sqrt{ab} = \frac{\log a + \log b}{2}$ pour tout couple (a, b) de nombres positifs.
- la fonction log est strictement croissante.
- Si $1 \leqslant a < b$, alors $a < \sqrt{ab} < b$.

Ces propriétés nous permettent de calculer une valeur approchée de log x pour tout nombre x compris entre 1 et 10.

Un Algorithme de résolution

Algorithme
Étape 1
Poser a = 1 et b = 10
log a = 0 et log b = 1.
Fixer ε ($\varepsilon = 10^{-n}$, par exemple)

Étape 2
Calculer $g = \sqrt{ab}$ et $\log g = \frac{\log a + \log b}{2}$

Étape 3
Si x = g, alors on pose log x = log g et on s'arrête.
Si x < g, alors on pose b = g et log b = log g
Si x > g, alors on pose a = g et log a = log g.

Étape 4
Si $b - a > \varepsilon$, on retourne à l'étape 2
Sinon on pose $\log x = \frac{\log a + \log b}{2}$ et on s'arrête.

Fin

Commentaire
On procède à une recherche dichotomique de x en comparant x à $\sqrt{ab}$. À la fin de l'étape 3, x est localisé à l'intérieur d'un intervalle plus petit que le précédent. On s'arrête quand la largeur de cet intervalle est inférieure à ε .

3.10.3 Résolution de l'équation f (x) = 0

Références: [20] , [48]

Utilisation
Math 101; Math 408

Analyse
Soit f, une fonction continue sur un intervalle fermé $[a_0, b_0]$. On suppose que $f(a_0)$ et $f(b_0)$ sont de signes contraires. On sait par le théorème des valeurs intermédiaires qu'il existe au moins un nombre x entre a_0 et b_0 pour lequel f (x) = 0.

Pour toute valeur donnée de ε, on peut calculer un intervalle de longueur inférieure à ε à l'intérieur duquel il existe au moins un nombre x tel que f (x) = 0.

Un Algorithme de résolution

Algorithme

Étape 1
Poser $a = a_0$ et $b = b_0$

Étape 2

Poser $x = \frac{a + b}{2}$ et calculer f (x)

Si f (x) = 0, on s'arrête.

Étape 3
Si f (x) et f (a) sont de même signe, poser a = x.
Sinon, poser b = x

Étape 4
Si b − a > ε , retourner à l'étape 2.

Sinon, $x = \frac{a + b}{2}$ est une solution au problème.

Fin

Convergence de la méthode
Après n itérations, la largeur de l'intervalle d'indétermination de x est égale à $\frac{b_0 - a_0}{2^n}$.

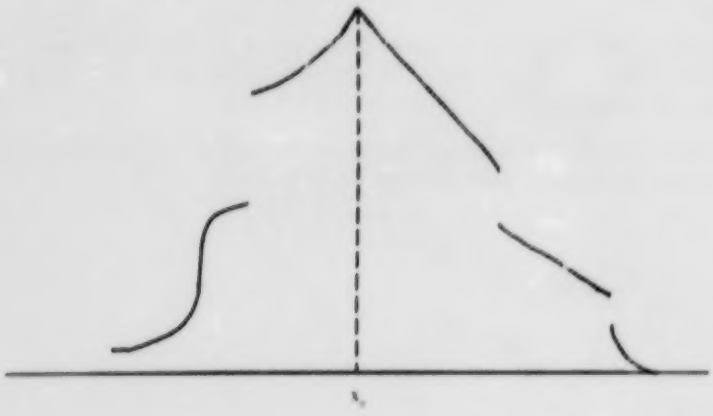
Remarque
Il existe bien d'autres méthodes pour résoudre ce problème.

3.10.4 Calculer le maximum local d'une fonction

Références: [20] , [52]

Utilisation
Math 408

Définition du problème
Soit f, une fonction réelle définie sur un intervalle fermé [a, b] de R. On ne sait rien de la différentiabilité ou de la continuité de f. On suppose que f possède un **maximum local unique** sur [a, b] , i.e. il existe un unique x_0 de] a, b [tel que f (x) ≤ f (x_0) pour tout point x d'un sous-intervalle] a_1, b_1 [de [a, b] , contenant x_0.



On se propose de calculer une valeur approchée de x_0.

Méthode par dichotomie
À une itération donnée, soit m le centre de l'intervalle d'indétermination de x_0 et soit m − h

et m + h, les extrémités de cet intervalle. Les points $m - \frac{h}{2}$, m et $m + \frac{h}{2}$ divisent $]m - h, m + h[$ en quatre sous-intervalles d'égale largeur.

x_0 se trouve dans l'un des trois intervalles

$I_1 = [m - h , m]$, $I_2 = [m - \frac{h}{2} , m + \frac{h}{2}]$, $I_3 = [m , m + h]$

- Si $f(m - \frac{h}{2}) > f(m)$, alors x_0 est dans I_1, à cause de l'hypothèse d'unicité du maximum local.
- Si $f(m + \frac{h}{2}) > f(m)$, alors x_0 est dans I_3.
- Si aucune des deux conditions précédentes n'est réalisée, alors x_0 est dans I_2.

Algorithme
Données: Le centre m de l'intervalle
La largeur 2 h de l'intervalle
la precision ε .

Étape 1
Calculer $f(m)$, $f(m - \frac{h}{2})$, $f(m + \frac{h}{2})$

Étape 2

Si $f(m - \frac{h}{2}) > f(m)$, poser $m = m - \frac{h}{2}$

Si $f(m + \frac{h}{2}) > f(m)$, poser $m = m + \frac{h}{2}$

Étape 3
Poser $h = \frac{h}{2}$

Si $h > \varepsilon'$, retourner à l'étape 1.
Sinon, poser $x_0 = m$ et $f(x_0) = f(m)$.

Fin

3.11 Recherche d'un arbre maximal dans un graphe

Référence : [3]

Utilisation

Math 109; Math 110

Définition du problème

Soit G = (V, E) , un graphe donné. V est l'ensemble des sommets du graphe et E, l'ensemble de ses arêtes. On suppose que G est connexe, c'est-à-dire que deux sommets quelconques sont reliés par une chaîne d'arêtes.

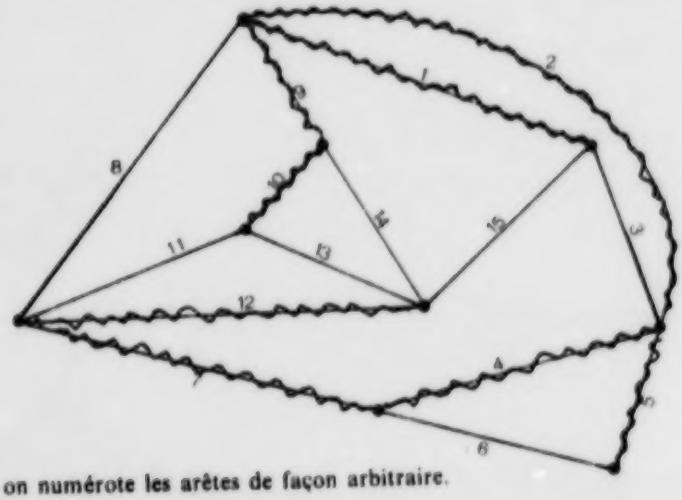
On cherche à construire un sous-graphe H de G qui soit un arbre, c'est-à-dire, un graphe H = (V, F) avec F⊂E, qui soit connexe et sans cycles (H est un arbre parce que connexe et sans cycles; il est de plus **maximal** parce qu'il recouvre l'ensemble des sommets de G).

Solution algorithmique

Algorithme
On choisit une arête e_1 quelconque.
On cherche une arête e_2 qui ne forme pas de cycle avec e_1; puis une arête e_3 qui ne forme pas de cycle avec $\{e_1, e_2\}$, etc... Quand la procédure ne peut plus se poursuivre, on a un graphe H = (X, F) , avec X ⊂ V et F ⊂ E. H est un arbre et X = V.

Justification
Il est évident que X ⊂ V et F ⊂ E.
H est un arbre parce que on peut caractériser un arbre comme étant un graphe sans cycle dans lequel un cycle (et un seul) se trouve créé quand on lui ajoute une arête supplémentaire ([3] , p. 22).
Finalement, X = V car sinon, on peut ajouter une arête à H sans créer de cycle.

Exemple



Au départ, on numérote les arêtes de façon arbitraire.

On choisit les arêtes 1 et 2 ; On rejette l'arête 3 ;
On choisit les arêtes 4 et 5 ; On rejette l'arête 6 ;
On choisit l'arête 7 ; On rejette l'arête 8 ;
On choisit les arêtes 9 et 10 ; On rejette l'arête 11 ;
On choisit l'arête 12 ; On rejette les arêtes 13, 14 et 15.

3.12 Recherche d'un arbre de poids maximum; Algorithme de Kruskal

Références: [3] , [24] , [33]

Utilisation

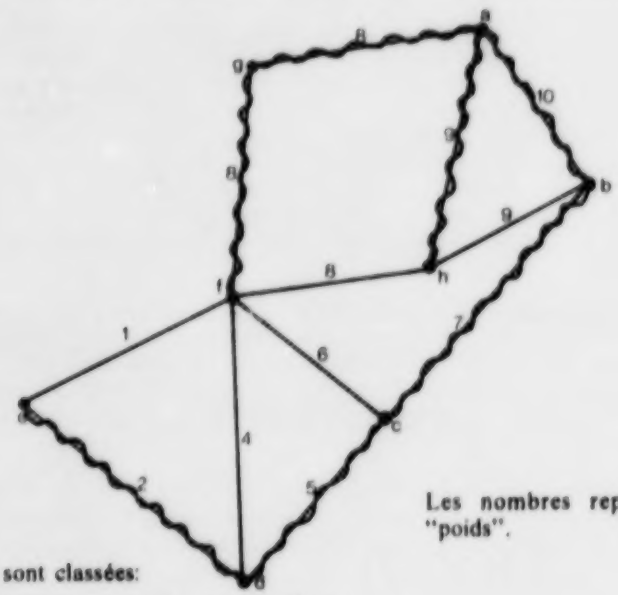
Math 109

Soit G = (V, E) , un graphe connexe donné. V est l'ensemble des sommets du graphe et E, l'ensemble de ses arêtes. On définit sur E une fonction p qui à chaque arête e associe un "poids", p (e) > 0. Trouver un arbre de G dont le poids est maximum.

Algorithme de Kruskal
Étape 1
Ordonner les arêtes de G par **ordre de poids décroissant.**

Étape 2
Choisir les arêtes une à la fois, dans l'ordre de poids décroissant. Une arête est rejetée quand elle forme un cycle avec les arêtes déjà choisies. On s'arrête quand toutes les arêtes ont été examinées.

Exemple



Les nombres représentent les "poids".

(i) Les arêtes sont classées:

[a , b] , [a , h] , [b , h] , [a , g] , [f , g] , [f , h] ,
[b , c] , [c , f] , [c , d] , [d , f] , [d , e] , [e , f] .

(ii) On choisit [a , b] et [a , h]. On rejette [b , h]
On choisit [a , g] et [f , g]. On rejette [f , h]
On choisit [b , c] On rejette [c , f]

On choisit [c , d] . On rejette [d , f]
On choisit [d , e] . On rejtte [e , f]

Le poids de l'arbre est égal à 49.

Remarque
La recherche d'un arbre maximal est un cas particulier du présent problème. Elle correspond à p (e) = 1, pour toute arête e du graphe.

Justification de l'algorithme de Kruskal
Voir section 3.13

3.13 Structure de matroïde; Algorithme "vorace" ou "glouton" ("greedy algorithm", en anglais)

Considérons le problème (P_1) de la recherche d'un arbre de poids maximum (section 3.12) et celui (P_2) de la recherche d'une famille de vecteurs linéairement indépendants de poids maximum (section 3.7). Ces deux problèmes appartiennent à des disciplines mathématiques différentes. Le premier d'entre eux est issu de la théorie des graphes tandis que le second provient de l'algèbre linéaire. Voilà pour les points de divergence! Maintenant, si on examine attentivement l'énoncé de ces deux problèmes et si on compare les deux algorithmes de résolution qui sont proposés, on constate qu'il s'agit en fait d'un problème et d'un algorithme uniques, énoncés dans deux contextes différents; on peut identifier les arêtes du premier problème aux vecteurs du second, les sous-graphes sans cycle aux systèmes de vecteurs linéairement indépendants.

On sait qu'en mathématiques pareille situation témoigne de l'existence d'une structure englobante plus générale. C'est ainsi, par exemple, que l'algèbre linéaire offre un cadre général pour traiter des objets mathématiques a priori aussi différents que les vecteurs géométriques, les matrices et les fonctions continues. De la même façon, la théorie des matroïdes offre un cadre conceptuel dans lequel on retrouve des propriétés communes aux graphes et aux espaces vectoriels.

Considérons un problème (P) qui s'énonce ainsi:

> Soit E un ensemble donné et soit $\mathcal{F}$, une collection de sous-ensembles de E. soit p, une fonction positive, définie sur les éléments de E. Déterminer un élément S de $\mathcal{F}$ tel que p (S) soit maximum.

Évidemment, (P_1) [resp. (P_2)] n'est autre que (P), exprimé dans le contexte de la théorie des graphes [resp. des espaces vectoriels] .

Supposons que $\mathcal{F}$ possède les propriétés suivantes:
a) $\emptyset \in \mathcal{F}$

b) Soit F , un élément de $\mathcal{F}$. Tout sous-ensemble de F est aussi un élément de $\mathcal{F}$.

c) Soit A, un sous-ensemble de E. Tous les éléments de $\mathcal{F}$ qui sont des sous-ensembles maxi-

maux de A ont le même nombre d'éléments (Un élément F de $\mathcal{F}$ est maximal dans A si aucun ensemble B tel $F \subsetneq B \subset A$ n'appartient à $\mathcal{F}$).

Traduisons ces propriétés dans le contexte de la théorie des graphes et dans celui de l'algèbre linéaire.

E = l'ensemble des arêtes d'un graphe G.	E = Un ensemble fini de vecteurs d'un espace vectoriel.
$\mathcal{F}$ = l'ensemble des sous-graphes sans cycle de G (chaque sous-graphe étant défini par l'ensemble de ses arêtes).	$\mathcal{F}$ = la collection des sous-ensembles de E qui forment un système linéairement indépendant.
a) $\emptyset$ définit un graphe sans arêtes, donc nécessairement sans cycle.	a) $\emptyset \in \mathcal{F}$
b) Tout sous-graphe d'un graphe sans cycle est lui-même dépourvu de cycle	b) Tout sous-ensemble d'un système linéairement indépendant est encore linéairement indépendant.
c) Si G' est un sous-graphe connexe de G, alors tout élément de $\mathcal{F}$, **maximal dans** G', possède n − 1 arêtes, où n est le nombre de sommets de G' ([3] , chapitre 3).	c) Soit E' un sous-ensemble de E. Tout élément de $\mathcal{F}$, **maximal dans** E' contient n vecteurs, où n est la dimension de l'espace engendré par E'.

Un couple (E, $\mathcal{F}$) ayant les propriétés a), b) et c) s'appelle un matroïde (voir [3] , [24] , [33] , [55]). Les matroïdes ont été définis par Hassler Whitney en 1935 pour généraliser la notion d'indépendance linéaire dans un espace vectoriel. On les rencontre en recherche opérationnelle et dans différents domaines des mathématiques.

Quand (E , $\mathcal{F}$) est un matroïde, le problème (P) possède une solution simple, donnée par l'algorithme suivant:

Algorithme Vorace

Données
Un matroïde (E, $\mathcal{F}$) et une fonction positive p, définie sur E. Soit n, le nombre d'éléments de E.
Résultats
Un sous-ensemble F de E qui appartient à $\mathcal{F}$ et qui est tel que p (F) est maximum.

Étape 1
Ordonner les éléments de E par ordre de poids décroissant. (Ils seront examinés l'un après l'autre dans cet ordre-là). Poser $F^0 = \emptyset$.

Étape 2
A la i^{e} itération, le i^{e} élément e_i de E est examiné. Soit F^{i-1} le sous-ensemble obtenu à l'itération précédente.

Si $F^{i-1} \cup \{e_i\} \in \mathcal{F}$, poser $F^i = F^{i-1} \cup \{e_i\}$.

Sinon, poser $F^i = F^{i-1}$

Si i est plus petit que n, retourner au début de l'étape 2. Si i = n, on s'arrête avec $F = F'$.

Fin

Justification

Théorème L'algorithme vorace produit un sous-ensemble de E qui appartient à $\mathcal{F}$ et qui est de poids maximum.

Avant de passer à la démonstration du théorème, nous devons établir un certain nombre de résultats.

1. La fonction p induit un ordre lexicographique sur les sous-ensembles de E. Soit
   $S = \{e_1, e_2, \ldots, e_k\}$ et $S' = \{e'_1, e'_2, \ldots, e'_l\}$.

   On suppose que dans S et S' les éléments sont numérotés par ordre de poids décroissant, c'est-à-dire:

   $$p(e_1) \geqslant p(e_2) \geqslant \ldots \geqslant p(e_k) \text{ et}$$
   $$p(e'_1) \geqslant p(e'_2) \geqslant \ldots \geqslant p(e'_l).$$

   Par définition,
   $S \succ S'$ si $k > l$ et $p(e_i) = p(e'_i)$ pour $i = 1, 2, \ldots, l$, ou bien s'il existe j tel que $p(e_i) = p(e'_i)$ pour $i = 1, 2, \ldots, j-1$ et $p(e_j) > p(e'_j)$.

   La relation $\succ$ est transitive. Un élément F de $\mathcal{F}$ est dit maximum au sens lexicographique ou $\succ$ – maximum s'il n'existe aucun élément F' de $\mathcal{F}$ tel que $F' \succ F$.

2. **Lemme** Si F et $F' \in \mathcal{F}$ et si F et F' ont respectivement k et k + 1 éléments, alors il existe un élément e de F' tel que $F \cup \{e\} \in \mathcal{F}$.

   Preuve du Lemme
   Soit $A = F \cup F'$. S'il n'existe aucun élément e de F' tel que $F \cup \{e\} \in \mathcal{F}$, cela signifie que F est **maximal** dans A.

   Ceci n'est pas possible. En effet, à cause de la propriété c), tout ensemble maximal dans A doit contenir au moins k + 1 éléments, ce qui n'est pas le cas de F.

3. **Corollaire**
   (i) Tout élément de $\mathcal{F}$ qui est $\succ$ – maximum est de cardinalité maximale.
   (ii) L'algorithme glouton produit toujours un élément $\succ$ – maximum.

4. **Preuve du théorème**
   Il suffit de montrer que tout élément $\succ$ – maximum de $\mathcal{F}$ est de poids maximum.

   Soit $F = \{e_1, e_2, \ldots, e_k\}$ un élément $\succ$ – maximum de $\mathcal{F}$. (On le suppose de cardinalité maximale).

   Soit $F' = \{e'_1, e'_2, \ldots, e'_l\}$, un élément quelconque de $\mathcal{F}$. Supposons

   $$p(e_1) \geqslant p(e_2) \geqslant \ldots \geqslant p(e_k) \text{ et}$$
   $$p(e'_1) \geqslant p(e'_2) \geqslant \ldots \geqslant p(e'_l).$$

Supposons que pour un indice i ($1 \leqslant i \leqslant \ell \leqslant k$), on ait

$$p(e_i) < p(e'_i)$$

Les sous-ensembles $H = \{e_1, e_2, \ldots, e_{i-1}\}$ et $H' = \{e'_1, e'_2, \ldots, e'_i\}$ appartiennent à $\mathcal{F}$, à cause de la propriété b). A cause du lemme, il existe $e'_j \in H'$ ($1 \leqslant j \leqslant i$) tel que $F = \{e_1, \ldots, e_{i-1}, e'_j\} \in \mathcal{F}$. On en déduit que $F \succ F$, ce qui contredit la $\succ$ – maximalité de F. Donc $p(e_i) \geqslant p(e'_i)$, pour $i = 1, 2, \ldots, \ell$.

Cela montre que $p(F) = \sum_{i=1}^{k} p(e_i)$ est maximum

Complexité de l'algorithme
Soit n le nombre d'éléments de E. Il faut vérifier n fois si un certain sous-ensemble de E appartient à $\mathcal{F}$. Tout dépend donc de l'aisance ou de la difficulté à effectuer cette dernière opération (voir section 3.3.7).

Chapitre 4

Quelques essais d'implantation de l'approche algorithmique

Il est arrivé au département de mathématiques du C.E.G.E.P. du Vieux-Montréal d'offrir un cours complémentaire axé sur la pensée algorithmique [43]. Cette expérience n'a pas pu se poursuivre longtemps, faute de moyens.
Cette année, dans le cadre de la présente recherche, l'approche algorithmique a été testée de façon informelle dans les cours de Math 105 et Math 205. L'accent y a été mis de façon délibérée sur l'analyse et la conception d'algorithmes: "Les algorithmes sous-jacents aux différentes méthodes utilisées seront mis en évidence, et là où il existe plusieurs algorithmes de résolution d'un problème, ils seront comparés en ce qui a trait à l'efficacité" [6].

L'enseignement dans ces cours s'est articulé autour de quelques idées simples:
— appel à l intuition
— rigueur dans les preuves et les analyses
— donner un sens à ce que l'on fait
— à chaque fois que l'occasion se présentait, relever les analogies ou les différences entre deux problèmes, deux résultats, deux méthodes...
— prendre soin d'insérer la matière étudiée dans le cadre des connaissances déjà acquises; faire un rappel de celles-ci, au besoin
— ...

Les cours se présentaient comme un laboratoire d'analyse et de résolution de problèmes: chaque théorème était vu comme un nouveau problème à la résolution duquel les étudiants étaient invités à contribuer. Souvent ils étaient mis en situation de "deviner" l'énoncé même du théorème.

Les commentaires des étudiants se trouvent résumés dans ce qui suit. (Quand deux opinions contraires s'expriment avec une certaine force, elles sont enregistrées toutes les deux).

Évaluation du Cours de Math 105 (Décembre 1983)

On a aimé

— la construction d'algorithmes; "un algorithme, c'est comme un "puzzle" ".
— l'enchaînement logique des différents chapitres et les liens qui les unissent
— pas de connaissances isolées
— le côté abstrait, "logique" (sic)
— différent des cours traditionnels de mathématiques
— bon pour la "logique" (sic), l'informatique

On n'a pas aimé

— trop de matière
— trop de théorie; trop de démonstrations de résultats abstraits; pas assez d'exemples numériques; pas assez d'exercices
— trop de notes

Suggestions

— avoir des notes écrites ou un livre avec des exercices
— meilleure répartition entre le contenu du cours et les exercices
— voir davantage de matière mais de façon moins détaillée

Interrogation

A-t-on vu assez de matière pour aborder et réussir Math 205?

Réflexion

L'adaptation au cours a été difficile.

Évaluation du Cours de Math 205 (Mai 1984)

On a aimé

— la présentation du cours; synthèses fréquentes
— bonne compréhension de la matière

On n'a pas aimé

— trop de matière
— présentation trop abstraite; manque de graphiques
— l'adaptation est difficile pour ceux qui ont suivi un cours de Math 105 exposé de façon traditionnelle
— pas de manuel de référence

Souhait

— faire suivre la théorie d'une séance d'exercices immédiatement.

Commentaires

— Dans les cours traditionnels on donne le résultat et on demande de l'appliquer; dans celui-ci, on part d'un exercice et on aboutit à un résultat
— Dans les cours traditionnels les démonstrations de théorèmes sont rares.

Bilan de cette expérience

Ce bilan informel est basé sur des impressions subjectives plutôt que sur une évaluation objective rigoureuse.

— L'adaptation a été très difficile. Mais ce qui frappe, c'est la participation soutenue des étudiants, du début à la fin. L'élément déterminant semble avoir été le fait que toutes les activités s'inscrivaient dans une structure de signification. Les étudiants participaient à la construction d'une chose qui leur inspirait crainte et respect mais qui avait perdu un peu de son caractère magique. En les mettant en situation de "deviner" l'énoncé d'un théorème à venir, on réconciliait mathématiques et gros bon sens à leurs yeux. En leur promettant "des larmes et du sang" dès la première séance de cours, on faisait appel à leur sens des responsabilités et cela a eu le don de stimuler leur effort.
— Le programme n'a pas pu être couvert en entier, loin s'en faut! C'est le prix à payer, semble-t-il.
— L'absence d'un manuel ou de notes de cours polycopiées a été durement ressentie.
— Dans l'ensemble, les étudiants ont pris goût à l'expérience et elle s'est révélée encourageante sur le plan des attitudes et de certaines habiletés: l'habileté à suivre un raisonnement, à analyser un problème et à le décomposer, l'habileté à intégrer différents éléments en un tout articulé...

Conclusion

On aura compris que l'approche algorithmique des mathématiques ne constitue pas une nouvelle recette magique, parée de toutes les vertus. Adopter cette approche ne signifie pas non plus qu'il faille bouleverser les structures existantes de fond en comble. Cette approche apporte une réponse à certains problèmes posés par l'enseignement des mathématiques et la liaison à établir entre cette discipline et l'informatique. Mais elle a toutes les choses d'être inopérante si, parallèlement, on ne consent pas un effort pour maintenir l'équilibre entre les différents pôles du développement cognitif et de l'activité mathématique (voir le chapitre 1).

Dans un plan d'implantation de l'enseignement des mathématiques du point de vue algorithmique, il faudra prendre en considération le contenu des programmes et le temps consacré aux activités de création d'algorithmes. L'analyse et la création d'algorithmes sont des activités qui consomment beaucoup de temps. Les programmes devront donc être repensés. Il faudra les alléger des éléments de connaissances devenus désuets et s'assurer de la présence des structures de connaissances nécessaires à la compréhension des algorithmes étudiés.

Grâce aux ordinateurs, les étudiants pourront vérifier l'exactitude de leurs algorithmes, effectuer des comparaisons, ... de façon expérimentale.

L'expérimentation sur ordinateur fait pendant à l'analyse théorique et représente un important enrichissement. Cela suppose la maîtrise d'un langage informatique. On aurait intérêt à utiliser un langage algorithmique, genre BASIC structuré ou PASCAL, qui conserve intacte la structure de l'algorithme et facilite d'éventuels changements et améliorations.

Si l'adoption de l'approche algorithmique semble répondre à certaines attentes, elle ne manque pas, à son tour, de susciter des questions. Faut-il adopter cette approche avant le

niveau collégial? Jusqu'où peut-on aller? Quid de la question des préalables? Quelles sont les conséquences à long terme d'un tel enseignement?

En cherchant une réponse à ces questions et à d'autres qui ne manqueront pas de surgir, on devrait être en mesure de préciser les conditions d'implantation de l'approche algorithmique et d'en établir une évaluation objective.

Références

(1) AHO, HOPCROFT & ULLMAN, **The Design and Analysis of Computer Algorithms**, Addison-Wesley, 1974.
(2) Howard ANTON, **Elementary Linear Algebra**, Wiley, 1973
(3) Claude BERGE, **Graphes et Hypergraphes**, Dunod, 1973 (2e édition)
(4) James K. BIDWELL, "Pascal's Triangle Revisited", **Mathematics Teacher**, May 1973, p. 448-452.
(5) BLOUIN, DAVESNE, GIARD, LALIBERTÉ, LAVOIE, **Algèbre Linéaire et Géométrie**, Gaëtan Morin éditeur, 1982.
(6) Jean-Marie BOURJOLLY, **Plan du Cours de Math 105**, C.E.G.E.P. de Sorel-Tracy, sept. 1983.
(7) Jean-Marie BOURJOLLY, **Notes de Cours d'Algèbre Linéaire**, en préparation.
(8) Wilfred. E. BOYKIN, "The Russian-peasant Algorithm: Rediscovery and Extension", **The Arithmetic Teacher**, Jan. 1973, p. 29-32.
(9) N. CALLEWAERT-BERCKMANS, "Calcul d'une intégrale définie à une approximation donnée', **Mathématique et Pédagogie**, n° 30, p. 53-57, 1981.
(10) Vašek CHVÁTAL, **Linear Programming**, W.H. Freeman and Company, 1983.
(11) Israël COHEN, "Pythagorean Numbers", **Mathematics Teacher**, Nov. 1974, p. 667-669.
(12) Commission Ministérielle sur l'enseignement des mathématiques, "Rapport préliminaire", **Bulletin de l'Association des professeurs de mathématiques de l'enseignement public**, 46e année, n° 258, 1967.
(13) John A. CRAWFORD, Calvin T. LONG, "Guessing, Mathematical Induction and a Remarkable Fibonacci Result", **Mathematics Teacher**, Nov. 1979, p. 613-616.
(14) George B. DANTZIG, **Linear Programming and Extensions**, Princeton University Press, 1963.
(15) H.B. DAVIES, "On highest common factor and least common multiple in the secondary school mathematics curriculum", **Int. J. Math. Educ. Sci. Technol.**, vol 11, no 1, 1980, p. 115-122.
(16) Leendert De LEEUW, "Teaching Problem Solving: An ATI Study of the Effects of Teaching Algorithmic and Heuristic Solution Methods", **Instructional Science**, Vol. 12, no. 1, p. 1-48, April 1983.
(17) Jean DIEUDONNÉ, **Algèbre Linéaire et Géométrie Élémentaire**, Hermann, 1978.

(18) Ronald EDWARDS, "Problem Solving Using GCD's and LCM's of Rational Numbers", **School Science and Mathematics**, vol. 80, 1980, p. 670-673.
(19) Arthur ENGEL, "The Probabilistic Abacus", **Educational Studies in Mathematics**, vol. 6, 1975, p. 1-22.
(20) Arthur ENGEL, **Mathématique élémentaire d'un point de vue algorithmique**, CEDIC, 1979.
(21) Arthur ENGEL, "Algorithmes et Calculateurs dans l'Enseignement des Mathématiques", **Tendances nouvelles de l'enseignement des mathématiques**, vol. IV, UNESCO, 1979.
(22) Arthur ENGEL, "Numerical Algorithms", **Mathematics Teaching**, No. 97, p. 43-46, December 1981
(23) Timothy A. GATHANY, "Involving Students in Problem Solving", **Mathematics Teacher**, Nov. 1979, p. 617-622.
(24) M. GONDRAN & M. MINOUX, **Graphes et Algorithmes**, Eyrolles, 1979.
(25) **Guide de l'UNESCO pour les professeurs de sciences**, Les Presses de l'UNESCO, 1981.
(26) R. GUNZENHÄUSER, "Computing in Mathematics Education", **Int. J. Math. Educ. Sci. Technol.**, vol. 6, n° 1, 1975, p. 17-26.
(27) R.W. JOHNSON, M. S. WATERMAN, "The Algorithms of Euclid and Jacobi", **Int. J. Math. Educ. Sci. Technol.**, vol. 7, n°3, 1976, p. 307-313.
(28) D.E. KNUTH, **The Art of Computer Programming**, vol. 1 (Fundamental Algorithms), Addison-Wesley, 1968.
(29) D.E. KNUTH, "Algorithms", **Scientific American**, p. 63-80, April 1977.
(30) **Konsèy Pou Kore Konesans Matematik** (Ideas for strengthening Mathematics Skills. Creole Edition), New York State Education Dept., Albany, Bureau of Bilingual Education, 1980.
(31) A. KRYGOWSKA, "L'enseignement mathématique au niveau postélémentaire (élèves de 10 à 16 ans), **Tendances nouvelles de l'enseignement des mathématiques**, IV, UNESCO, 1979.
(32) Lyman HOLDEN, "A Classroom Note on the Euclid Algorithm", **School Science and Mathematics**, vol. 74, 1974, p. 698-700.
(33) Eugene LAWLER, **Combinatorial Optimization: Networks and Matroids**, Holt, Rinehart and Winston, 1976.
(34) Pierre LEROUX, **Algèbre Linéaire. Une approche matricielle**, Modulo Editeur, 1983.
(35) MATHECRIT, **Ateliers 105. Calcul Vectoriel et Matriciel**, 1976.
(36) George McCABE Jr., "Alternate Methods for Finding LCM and GCF", **Mathematics Teacher**, Jan. 1979, p. 34.
(37) G. MEANT, "Addition et Soustraction dans une base quelconque", **Mathématique et Pédagogie**, n° 30, p. 5-15, 1981.
(38) George MIEL, "Of Calculations Past and Present: The Archimedean Algorithm", **American Mathematical Monthly**, vol. 90, n° 1, 1983, p. 17-35.
(39) Vedula N. MURTY, Frank J. SWETZ, "Determinants, Diagonals and Discovery", **Mathematics Teacher**, Nov. 1982, p. 682-685.
(40) John NIMAN, "Graph Theory in the Elementary School", **Educational Studies in Mathematics**, vol. 6, 1975, p. 351-373.
(41) Ben NOBLE & James W. DANIEL, **Applied Linear Algebra**, Prentice-Hall, 1977.
(42) David L. PAGNI, "Number Theory for Secondary Schools?", **Mathematics Teacher**, Jan. 1979, p. 20-22.
(43) Richard PALLASCIO, Communication personnelle, 1983.
(44) Howard A. PEELLE, "Euclid, Fibonacci and Pascal — recursed", **Int. J. Math. Educ. Sci. Technol.**, vol. 6, n°4, 1975, p. 395-405.
(45) D.A. QUADLING, "L'enseignement des mathématiques dans le second cycle de l'enseignement secondaire (lycées et établissements assimilés)", **Tendances Nouvelles de l'Enseignement des Mathématiques**, vol. IV, UNESCO, 1979.
(46) Jane I. ROBERTSON, "How To Do Arithmetic", **American Mathematical Monthly**, vol. 86, n°6, 1979, p. 431-439.
(47) Twila SLESNICK, "Algorithmic Skill vs. Conceptual Understanding", **Educational Studies in Mathematics**, vol. 13, no. 2, p. 143-154, May 1982.
(48) Stephen L. SNOVER, Mark A. SPIKELL, "Generally, How Do You Solve Equations?", **Mathematics Teacher**, May 1979, p. 326-336.
(49) Robert G. STEIN, "A Way To See $1^2+2^2+...+n^2$", **Mathematics Teacher**, vol. 67, Nov. 1974, p. 638.

(50) Mirette TORKIA-LAGACÉ, **La pensée formelle chez les étudiants de Collège I: Objectif ou Réalité?**, Rapport final d'une recherche subventionnée par PROSIP, Cégep de Limoilou, 1981.
(51) Stewart M. VENIT, "Approximate Integration, Comparative Examples", **Mathematics Teacher**, Dec. 1978, p. 774-775.
(52) Harvey M. WAGNER, **Principles of Operations Research**, Prentice-Hall, 1975 (2e édition).
(53) William M. WATERS Jr., Edward G. BLAKEWAY, "The Old and the New: An Ancient Egyptian Algorithm and the Computer", **School Science and Mathematics**, vol. 76, 1976, p. 200-202.
(54) Cloman WEAVER, "Figurate Numbers", **Mathematics Teacher**, Nov. 1974, p. 661-666.
(55) H. WHITNEY, "On the abstract properties of linear independence", **Am. J. Math**, 57, 1935, p. 507-553.
(56) Herbert S. WILF, "The "Why-Don't-You-Just" Barrier in Discrete Algorithms", **American Mathematical Monthly**, vol. 86, n°1, 1979, p. 30-36.
(57) N. WIRTH, **Algorithms + Data Structures = Programs**, Prentice-Hall, 1976.

I.S.B.N. 2-550-07591-9

89-0042